Developing Appiducks - an app for learning Python

Solveig Raddum

Supervisor: Michal Walicki

Department of Informatics
University of Bergen

August 2019

Acknowledgements

I would like to thank my supervisor Michal Walicki for guiding me through the process of writing this thesis. A special thanks goes to my co-supervisor Anna Maria Eilertsen for valuable feedback early in the process of this thesis. A thanks goes also, to May-Lil Bagge for all the work she has done with the requirement - and questions documents for the app, AppiDucks.

Abstract

The last couple of decades apps made for smartphones have become a big industry. People use apps in many areas in life, like entertainment and learning. In app development one of the challenges is that an app developed for one device's operating system (OS) can not easily be used on a device with another OS. In this master thesis we have looked at different approaches for developing apps. One is the *native* approach, which gives an app developed for one particular OS, and the other is *cross-platform* solution, that can run on more than one OS. We evaluate the pros and con with the different approaches for developing an app for learning Python.

We implement a prototype of the app with a cross-platform approach using Google's new cross-compiled framework, Flutter. This is to see if you get an app with one code base that can run on devices with both Android or iOS operating systems. The final prototype of the app runs on both OS without needing any customization of the app's code. The prototype of the app was then used to do user survey among students at UiB. The user survey was done to get knowledge about the effect the learning app has for learning Python.

Table of contents

Chapter 1

Introduction

The technology development has changed the way we live our lives in many areas. From a time where we had to personally show up in the bank and stand in a line to pay our bills, via being able to do it from a stationary PC in our living room, to now where we can do it on the go from an app on our smartphone.

The introduction of smartphones happened with Apple introducing iPhone with a touch-screen to the market in 2007 [14]. About a year later apps could be installed through App store on iPhones with the iOS operating system (OS). The iPhone quickly became popular and other companies like Google and Microsoft introduced their own smartphone devices with their own OS, Android and Windows.

With the popularity of the smartphone the app development has become a big industry over the past decade and we can find an app for almost everything. Apps have changed how we interact with each other and the way we live our lives. In education, learning apps have become a new way of learning on the side of the traditional education institutions. The learning apps give the learners a possibility to take learning with them wherever they go. The learning is no longer so connected to time and place as it used to be.

There have been created frameworks which provide a platform for developing apps. This has made it simple for everyone to make an app without much development experience and the app stores make it relatively easy to get the finished app to the market.

That the largest category at Google Play is education [13], shows that the opportunity to learn by apps has become a popular way of learning. With more than 200 000 apps in the learning category at Google Play it can be hard for the user to find a good learning app. This is especially true since there are no requirements for what can be called a learning app. Today most of the learning apps have been developed without education experts involved in the process and therefore there is no guarantee for the educational content in the app

[31]. A learning app is still a new way for learning and research has to be done to get more knowledge about how to create a quality learning app.

Even though it is easy find and install an app, the app technology is not a uniform technology. iPhone, Android and Windows devices' operating systems (OS) are different from each other. The result of this is that an app made for iPhone devices' OS, a native app, is not compatible with a device with another operating system. It is also an area in constant change as new devices or new versions of devices are put on the market on a regular basis. This makes it difficult to develop apps that look and behave the same way on all the different devices.

Developing an app for all the different OSs demands expert knowledge of more than one application programming interface (API), integrated development environment (IDE) and programming language from the developer and it will also be more expensive and time consuming. It will therefore be useful to consider if you really need to develop a native app for each OS or if you can go for a cross-platform solution, an app compatible with more than one operating system.

1.1 Motivation

The new generation of students which now start at higher education institutions have had smartphones and apps as a part of their everyday life from an early age. This, together with the popularity of learning apps, make it likely to think that they will try to find apps that can help them in their study.

The use of a learning app as a tool for learning is quite new and has not been a part of the traditional education. The app stores are still more like a commercial market, where everyone can make an app and categorise it as a learning app for making money.

The learning apps' popularity makes it necessary for the traditional education system to start to look at how effective apps are for learning. What makes a good learning app? How to best implement a learning app for use in the education? These are questions we need to get more knowledge about before the education institutes can start to use learning apps in addition to the regular teaching.

A learning app developed for use in the education will make it easy for students to find quality apps with content that is relevant for the course they are following. This, together with more knowledge about the effect of using such an app, will give the students an awareness about whether using the app will help in learning the subject or if it will be wasted time.

This master thesis does not focus on a specific research area. It is more practically directed where the aim has been to look at different approaches in app development and find

a suitable framework for developing an intended learning app called AppiDucks, for learning Python. The final app is thought to be used in user survey to evaluate its effectiveness of using learning apps as a supplement to the normal education in INF100, Introduction to programming, at the University of Bergen (UiB). INF100 is the first course in programming at UiB.

After deciding that we will go for cross-platform as the technology solution for AppiDucks, we look at two different technical options for frameworks in order to develop a cross-platform app. One is Ionic, which is a hybrid framework where you use web technology to develop the app, and the other is Flutter, that is a cross-compiled solution where the code you write gets compiled to native code that is run on the devices.

With the cross-compiled framework Flutter, a prototype of the app is implemented with questions from 3 weeks of the syllabus of INF100. After the prototype has been implemented a small user survey is done among students following INF100 in the spring of 2019 to get some data on the effect of using AppiDucks as a supplement to the traditional education of INF100 at UiB. The result from the user survey is evaluated and used as a base for the future development of AppiDucks.

1.2 Organization of the thesis

Chapter 2 Apps for Use in Education

In this chapter we describe the requirements for a learning app for Python, and compare those with an existing learning app for Python.

Chapter 3 Background on App Development

This chapter is about app development, where we look at the different types of apps and operating systems. The difference between cross-platform solutions and native apps is studied, where we in the first part of the development process look at the different solutions for developing apps. These are native, web, and hybrid mobile apps. We evaluate the pros and cons regarding the different types of apps for our project.

Chapter 4 Developing Appiducks

In Chapter 4 we describe how we have implemented the requirements for the intended app Appiducks, and made a prototype with 3 weeks of questions from the syllabus of INF100.

Chapter 5 User survey of the app

In Chapter 5 the execution of a user survey among INF100 students is described, and the results of it are discussed.

Chapter 6 Conclusion

In chapter 6 the results of the master thesis are summarized, together with the possibilities for future work.

Chapter 2

Apps for Use in Education

In this chapter we describe the requirements for a learning app for Python provided by an administrative assistant for INF100 Autumn 2018, and compare those with an existing learning app for Python.

2.1 Education today

The last couple of decades there has been a huge change in the traditional education as the technology has developed and become more integrated in it.

Today most learning institutions have implemented the concept of Bring Your Own Device (BYOD), which means that people bring their own devices such as laptops, smartphones, etc. with them to use in the learning environment [9].

Mobile learning is in [27] defined as "any type of learning that takes place in learning environments and space that take account of the mobility of technology, mobility of learners and mobility of learning". With mobile communications network we get more access to individualized learning than the previous PC-based platform. [21].

As technology continues to grow it will continue to be integrated in education. In [8], learning analytics is defined as "the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimising learning and the environments in which it occurs." Adaptive learning is defined "as a way of delivering learning materials online, in which the learner's interaction with previous content determines (at least in part) the nature of materials delivered subsequently" [22]. BYOD together with learning analytics and adaptive learning is expected to grow in the years to come [9]. Today, most of the students own a smartphone which they bring with them almost everywhere. By making learning apps for the different courses that the education institute offers, they can use smartphones as a part of mobile learning and BYOD. Learning apps for smart phones

can give university students another way to learn outside of the classroom. In [32] the language students who used a learning app tell in the survey that they found it "easy to use and understand" and "accessible anywhere, anytime". A learning app will also give the education institution an opportunity to gather data from students through the online learning environment.

Apps in use at the University of Bergen

Institutions are increasingly designing apps to allow students access to administrative and educational content [21]. MittUib and ArtsApp are two apps used in education by the University of Bergen (UiB), which are both categorized as educational in Google Play. They show up together with some other apps that are made to make the student life at UiB easier, apps likes "StudentBevis" and "Romplan for UiB", when you use 'UiB' as search words in Google Play.

The app MittUiB is described in Google play to be an app with administrative content and is used to communicate with the students. The students download the app and it gets its content from the learning platform mitt.uib.no about the courses the student has signed up for. The smart phone makes all the relevant information for students available any time and everywhere through the app making it personalised for each student.

While MittUib is more an app with administrative content, ArtsApp is an app used by biology students in a learning situation. At the Department of Biology at UiB the students and the lecturer have designed ArtsApp. The app acts as an interactive identification key to help students with species identification. The app is adapted to what the students and the lecturer think will make it easier to identify species. They have made it more flexible by letting the users choose which characteristics they prefer in the identification of a species instead of forcing them to use a dichotomous key in the identification. That the app allows the user to choose their preferred identification characteristics and to save their observations, make both the content and the learning more personalised [3].

2.2 Apps for use in education

Google Play has more than 2 600 000 apps for Android phones. The statistics from AppBrain [13] show that out of the ten most popular categories, the education category is number one with 229 422 apps (18.04.2019), where 6 percent have been downloaded more than 50 000 times.

With so many apps in the learning category what can a learning app developed for a particular course give the student that can not be found in an app that already exists?

A study of undergraduate students in technology by the Educause Center for Applied Research (ECAR) showed that learners' use of mobile technology has risen substantially in the last decade[21]. With 229 422 learning apps in Google Play [13] and about 200 000 in Apple's App Store [12], it is natural to think that new students will try to find a learning app that can help in their studies. It will therefore be important that the app they choose has a certain level and quality. This is something that will require that the students have a sufficient pre-knowlegde in the field to make a decision if the app content covers the syllabus properly to help them in the preparation for an exam.

We should take into concern that determining the quality of an app is hard because of the lack of research in that area, together with the fact that the information in App Store and Google Play is supplied by the developers and therefore is more informative than about educational content in the app [25]. This is something that can make it difficult for students to know if an app has good enough quality to use as a supplement in the learning process at a higher education institution.

A learning app developed as a supplement for a particular course will have a content that reflects the syllabus for that course. It will also emphasize what the teacher thinks is important. The app's content will be built up in a way that is pedagogical and will be updated in the learning app if the course changes. By creating a learning app for a course the teacher gets to determine the overall learning goal for the whole app and for each lesson in it. When a student has finished a lesson, the student gets feedback about what she knows and what she needs to work more with.

By using a learning app the teacher can collect information through the semester from the app about how the students answer the tasks in the app. This gives the teacher an opportunity to see how the students learn the syllabus and adjust the education at an early stage if needed.

2.3 The intended app: AppiDucks

AppiDucks is thought to be used as a supplement to the normal classes and group education in INF100. The app gives the students an opportunity to take the learning outside the classroom and to learn on the go.

The student chooses a week from the start page. A lesson with twelve questions starts with the first question displayed on the screen. The user answers the question and a feedback shows up on the screen and tells if the answer is right or wrong. The feedback for wrong answer also tells what the right answer is. The user then clicks the next-button on the page and the next question is displayed on the screen. The student answers the new question and clicks the next-button and a new question shows up on the screen. This continues until the 12

questions in the lesson are answered, and the summary page is displayed with information on how many correct answers the student managed and a button to get back to the start page where the student can start an new lesson if he or she wants to.

2.4 Requirements for AppiDucks

AppiDucks will not be a standalone training for learning Python. It is made as a supplement to the twelve weeks INF100 course in Python. The requirements for the prototype of the app are given by the document "App for å lære" by May-Lill Bagge (see Appendix A). This document describes the whole intended learning and the requirements for how it should be. The prototype of the app has used the following requirements from the document in the implementation.

- Twelve lessons which reflect the twelve weeks with syllabus of INF100
- One lesson which can be seen as exam preparation
- Each lesson has a subset of 12 questions from its corresponding week.
- The lessons have incorporated positive feedback in them. The feedback is given after 3, 5, 10 and 12 correct answers in a row
- The students can use as much time he or she wants to answer a question
- The answer is either right or wrong
- The app gives feedback for each answer, if it is right or wrong
- if the given answer is wrong, the student gets in the feedback what the right answer is.
- After each lesson the student gets to know how many right answers they had
- The app should be developed as a cross-platform app

Requirements for the twelve weeks lessons

Each of the twelve weeks have separate documents that describe requirements for each week's lesson and questions (see Appendix A, B,C, D and E). The prototype has three weeks with questions implemented. Each week's lesson has general questions that cover the theory from that week's syllabus. The implementation of general questions are described in Section 4.3.1. Below, in 2.4.1-2.4.3, we summarize the requirements for the specific questions of the three first weeks.

2.4.1 Requirements for week 1

These are the requirements of the types of questions from week 1 that lesson 1 contains, in addition to the three general questions. These are taken from the Document "Uke 1" by May-Lill Bagge (Appendix B).

1. Three math questions.
   Tasks to practice with simple math operations like division, integer division, floor, modulo and power. (see Appendix B pages 29-32)

2. Three precedence questions.
   Tasks to practice with operator precedence and how parentheses influence the order the operators will be executed (see Appendix B pages 33-44)

3. One question about type.
   Simple task to get the user familiar with the integer and the float types. (see Appendix B pages 32-33)

4. One question about argument and assignment.
   Simple task just to see that students understand assignment (see Appendix B pages 44-47)

5. One question about augmented assignment.
   Simple task which gives the student practice in evaluating and calculating augmented assignments (see Appendix B pages 47-53)

The implementation of the week's question types are described in Section 4.3.2.

2.4.2 Requirements for week 2

The idea for the requirements to the questions for week 2 are described in the documents Uke 3 og 4 for Python App by May-Lill Bagge (Appendix C and D). From those documents the following questions are implemented in addition to two general questions.

1. Two questions about string lengths.
   Tasks for practicing use of the built-in len-function in Python

2. Three questions about concatenation with two strings.
   Tasks for practicing concatenation with strings

3. Two print questions.
   Tasks for practicing the use of print-functions

4. Two questions with concatenation of the string and integer types.
   Tasks for practicing concatenation with different types to see who they work

5. One question about concatenating three strings.
   Task for practice concatenation with strings

The implementation of the questions for week 2 is described in Section 4.3.3.

2.4.3 Requirements for week 3

A lesson for week 3 consists of the following questions, in addition to three general questions. These are taken from the document Uke 2 av May-Lill Bagge (Appendix E).

1. Four questions with built-in functions in Python.
   Tasks to practice simple use of Python's built-in functions. (see Appendix E, pages 18-27)

2. Two questions with missing code parts.
   Tasks where the user has to see what is missing in a small code part (see Appendix E, pages 27-40)

3. Two questions about what a code returns.
   Tasks for understanding a small code part (see Appendix E, pages 41-52)

4. One question about error-messages.
   Task for understanding a small code part (see Appendix E, pages 56-59)

The implementation of the third week's questions are described in Section 4.3.3.

2.5 Comparing the requirements of the intended AppiDucks with SoloLearn Python

A search in Google Play with the term "learn python" gave more than 287 available apps. On the top of the list was SoloLearn Python, a learning app for Python which has been installed 100 000+ times. SoloLearn Python was also suggested to me by a student that had been a teacher assistant in INF100 in the autumn 2018 to be suitable for comparison with AppiDucks.

SoloLearn Python is built up of 9 modules. Each module has a name that explains which concept from Python programming it covers. The nine module names are: Basic Concepts,

Control Structures, Functions & Modules, Exceptions & Files, More Types, Functional Programming, Object-oriented Programming, Regular Expressions and finally Pythinicness & Packaging. When you start to learn Python with SoloLearn Python only the first module, basic Concepts, is open. You have to finish one module to open the next one. The user is given an opportunity to take a short cut and skip the 3 first, 6 first or the 7 first modules. The short cut is a test with 10 questions. The user can only have 3 wrong answers to pass the short cut and open the module after it. When you have done 9 modules you get a certificate.

SoloLearn Python is a stand-alone tool to learn Python on a smartphone or a computer. AppiDucks, on the other hand, is not a stand-alone tool for learning Python. AppiDucks is only available on smartphones and the app is thought to be used by students at UiB taking INF100. Each week with questions in AppiDucks reflects what is covered in each of the 12 weeks of syllabus in INF100.

With SoloLearn Python the user learns the theory in Python as he or she does the lessons in the modules. Each lesson starts with a small theory part, around half a page, about the subject from Python the lesson is covering. The questions that test if the user has understood the theory are found on the next pages.

When a user has answered, she or he only gets feedback if the answer is right or not. If the answer is wrong the user has to answer the question again until he or she gets it right before moving to the next question. If the user really struggles it is possible to get help in the form of a hint. In SoloLearn the lessons are short. The app has adaptive learning since the user has to finish one module to get to next one.

SoloLearn also offers users communication with each other, by asking questions or making comments about Python.

AppiDucks for Python does not have any theory in it since it is meant to be a supplement to the course INF100. The app should not be used as an independent learning app, but as an addition to the traditional education given at UiB. The lessons in AppiDucks can be seen as the short-cut tests in SoloLearn Python.

When a student takes a lesson she or he practices what they have learned at the lectures. At the end of each lesson the user gets feedback of how many correct answers she or he had in that week's syllabus. The prototype of AppiDucks does not have adaptive learning in it. All twelve lessons in AppiDucks are open. This is to make it easy for the students to start using the app in the middle of the semester. If a user of AppiDukcs gives a wrong answer, he or she is told what the right answer for the question is before the next question.

It is not only the students that get feedback on what they have learned and not. The teacher can also pay attention to the results, and if some parts seem to be problematic with many of the students, the teacher can adjust the next lecture accordingly.

The students that use AppiDucks can not communicate via the app, but since Appiducks is a supplement to a course at UiB, the students have access to groups where they can go and discuss problems. A lesson in AppiDucks lets the user practice more than one topic at the time, while in SoloLearn a lesson has focus on one topic at the time, except from the last lesson in a module which acts more as a test that covers all the lessons in the module.

As a learning app for python, AppiDucks will add something new as its content is based on the syllabus of INF100. It is also different from other learning apps in the way that you should follow the course to use the app since the app has no theory in it. It is an app that will give the students an opportunity to take a test of what they have understood of the week's syllabus. At the same time they take the test, they get to practice on tasks from the syllabus.

It will therefore be of interest to develop a prototype of the app and have a user survey to see if the app has an effect on learning Python for students following INF100.

Chapter 3

Background on App Development

In this chapter we look at two different solution for developing apps. Those are: native and cross-platform apps, and the main operating systems for apps, iOS and Android. We evaluate their pros and cons regarding the two different approaches for developing an app. Then we decide what is the best solution for developing appiDucks in our project.

3.1 Native apps

The industry of developing mobile apps has grown a lot the last couple of years, with more than 5 million apps available throughout different mobile app stores in 2017 [16]. The largest companies are Android's Google Play (2.8 million apps) and iOS' Apple App Store (2.2 million apps) [16].

A native app is an app developed for a specific mobile platform, such as Android or iOS. Each of the different platforms comes with tools and a development environment for developing native app for the platform. The native app has to be written in the platform's official language and will not be able to run on another platform.

3.1.1 iOS

Apple has developed the iOS Software Development Kit (SDK) for developing mobile apps for Apple's iOS operating system [38]. The SDK gives the developers access to diverse functions and services of iOS, such as hardware and software attributes of the devices. It comes with an iPhone simulator. The simulator acts as a device on the computer so the developer can see how the app looks under development of the app, without having to deploy the app to a physical device. In addition to iOS SDK, developers use Xcode as an integrated

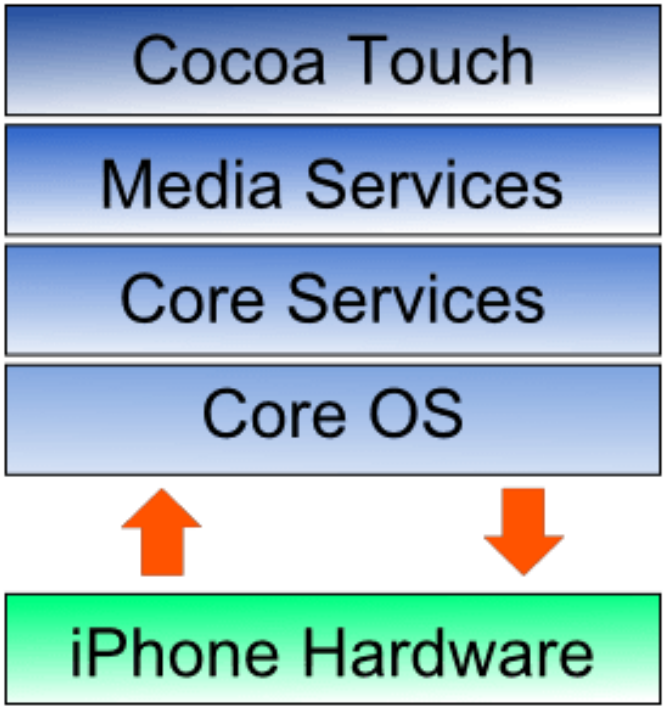


Fig. 3.1 iOS Architecture [33]

development environment (IDE) and write the app with the official software languages Switch and/or object-C.

On iPhone, Apple does not let the developer of apps get direct access to the hardware within the iPhone. The architecture of the iPhone's OS is built up of software layers (see Figure 3.1). Those layers act as intermediaries between the app and the hardware that comes with the iPhone [33]. Each of the layers consist of different frameworks the developer can use when developing the iPhone app.

- Cocoa Touch layer - this is the top layer and contains the framework that is most used by the app developer
- Media Services layer - contains the frameworks that provide the iPhone with audio, video, animations and graphics
- Core Services layer - has the framework that gives a foundation for the above layers
- Core OS layer - is on the top of the device hardware. This layer provide services to low level networking, access to external accessories and fundamental operating system services.

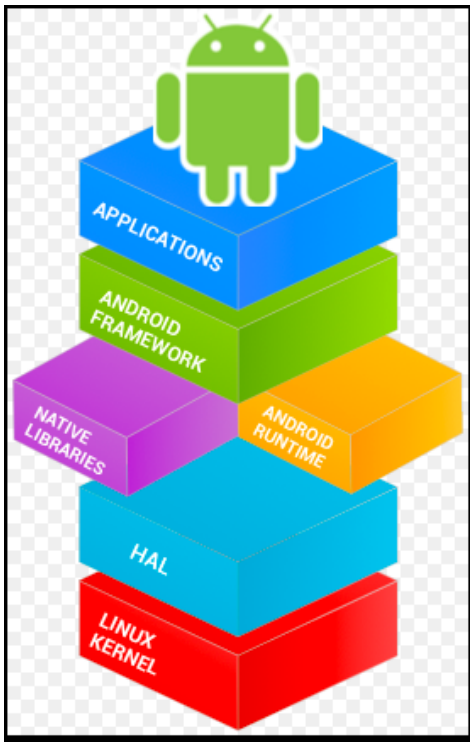


Fig. 3.2 Android Stack[40]

3.1.2 Android

Android is an open source, linux-based software stack [10]. Figure 3.2 shows the major components of the Android platform. Android's IDE for developing mobile apps is Android Studio and the official languages are Java and Kotlin. Android Studio gives the developer access to Android SDK. The Android SDK makes it easier to run the app on Android devices and to utilise hardware components.

In the Hardware Abstraction Layer (HAL) you find standard interfaces that expose the device' hardware capabilities to the higher-level Java API framework. This means that when the programmer makes a call through the Java API framework for access to a devices' hardware component, such as the camera, the Android system loads the library module for that hardware component from the hardware abstraction layer.

The Java API Framework offers you the feature set of Android OS as building blocks through an API written in Java. The developer can reuse the core, modular system and services as

- View system for building the UI for the app
- Resource Manager

- Notification Manager
- Activity Manager that manages the app's life cycle and provides a common navigation back stack
- Content Provider, which gives the app promotion to share their own data or access other app's data

3.1.3 Advantages and disadvantages of native apps

Native apps built for a specific platform make full utilisation of all the device's features. This is the reason native apps provide the best solutions when it comes to user experience and performance [23]. The native apps will also be easily available through the different platforms' app stores.

Developing native apps for all the different platforms requires knowledge about the programming language, Application Programming Interfaces (API) and Software Development Kits (SDK) of each target platform, which demands additional resources (skills, time and cost) [15]. This is something that is a huge disadvantage when it comes to developing native apps.

Various approaches have been taken to try to find a solution for solving the problem that a native app can only be distributed for one platform. In the last couple of years, attention has been given to making a framework for cross-platform development [19]. Cross-platform solutions give the developers an opportunity to write one application which can be deployed to all mobile platforms supported by the framework that is used to create the app.

For learning apps a cross-platform solution will be a good solution since it is time consuming to develop apps. It will also be too expensive to make native apps for all different platforms, according to [20].

Also, for the prototype of the learning app appiDucks, a cross platform framework will be the best solution, since the app does not use any of the built-in features that come with the device. Therefore it will not need to be implemented as a native app to get an optimal performance. The choice of a cross-platform framework will also be more time saving since we only have to write the code once.

3.2 Cross-platform development

Cross-platform development is a term used for many different concepts: technology, approaches, frameworks and libraries, where each approach comes with pros and cons. The

frameworks for a cross-platform deployment are divided into four different approaches: The web approach, the hybrid approach, the interpreted approach and the cross-compiled approach [19]. What they all have in common is

- common code base
- saving time and money
- can be used on the different platforms

Two potential frameworks were considered for implementing AppiDucks. The hybrid mobile framework Ionic, which was released for the public in 2013 with the last stable version from November 2017 [36], and the new cross-compiled framework Flutter from Google which was released as Preview 1 in July 2018 [35]. Apps written with Ionic will be available for both iOS, Android and Windows. Flutter apps on the other hand are only available for the iOS and Android platforms.

3.2.1 Ionic

Ionic is a hybrid framework with almost 4 million apps created by using it since 2014 [11]. The Ionic framework uses web technologies including HTML, CSS and JavaScript together with Apache Cordova. Apache Cordova is a tool and library for initialising a new hybrid app. It uses a command-line interface to generate a new native app with a WebView and two-way communication between the WebView and native code. Cordova provides plugins to get access to resources such as camera and GPS [16].

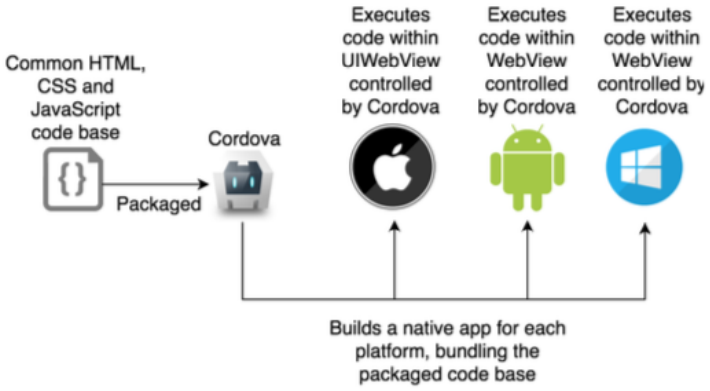


Fig. 3.3 Overview of the hybrid approach build framework [16]

In Ionic the user interface is built up with HTML, CSS and JavaScript. The logic in an app created using the Ionic framework is written in AngularJS. AngularJS is a JavaScript-based

open-source front-end web application framework. To get the UI components to look like native, Ionic uses Angular.js directives. This gives non-standard HTML tags for rendering the interface components.

Ionic does not offer any test environment with the framework. In [26] it is recommended to use the frameworks Jasmine and Karma for testing, where Jasmine [37] is a framework for writing behaviour tests and Karma [41] is the framework for running the tests.

Installing Ionic

For Ionic you have to install some utilities. On [1] you will find what you need, and it looked pretty straight-forward to install. However, when we first tried it, we were directed to install Ionic version 1 which is an older version. With Ionic the installation process was confusing since there were so many utilities and some of them depended on other components. In the end it was difficult to know if you had got the right version and if you really had everything you needed.

When you open a new project in Ionic you get a template of an app project. This gives you an opportunity to see how the folder structures of a project are. You can view how the layout is on different devices in a web page at the same time, see Figure 3.4.

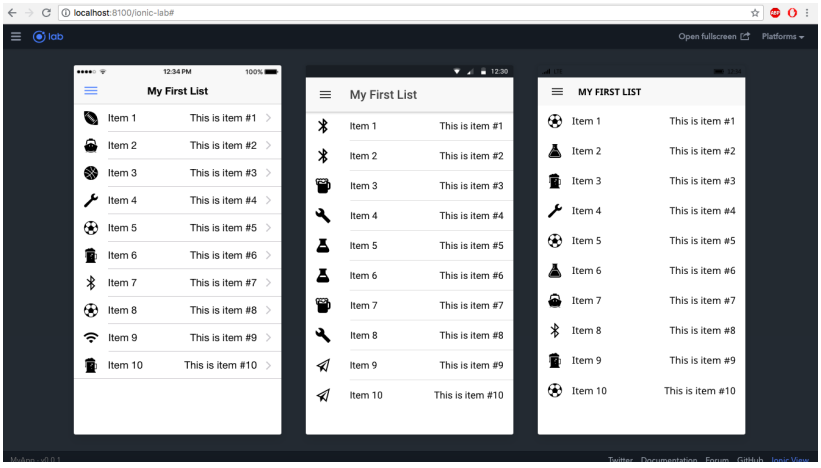


Fig. 3.4 The figure shows how an Ionic app built from a template looks on different platform devices

Just to get some experience one may try out creating a new page and pass data between them, but for Ionic it was hard to find out how to do it. For understanding how to get started with the framework on your own, you need knowledge not only about HTML, CSS and javascript/typescript. From searching for answers to problems when developing with Ionic we found that also understanding AngularJS is necessary if you want make use of the properties and functionalities of the framework. Quoting from Jeremy Wilken [39]:

> If you don't know AngularJS yet, that is ok! However you do need to be prepared to learn it. To make a top quality app, you will have to be able to leverage Angular to its fullest.

So it was a steep learning curve and the framework was absolutely not self-explanatory. Ionic was hard to understand and not easy to get started with from scratch from our background.

3.2.2 Flutter

Flutter is created by Google [4]. It is an open-source mobile application development SDK to write apps for Android and iOS [35]. The Flutter framework uses a cross-compiled approach. In the cross-compiled approach a common language is used that can be compiled to native byte code that runs on the different platforms, see Figure 3.5. Flutter also has plugins for iOS and Android to get resources such as the device cameras.

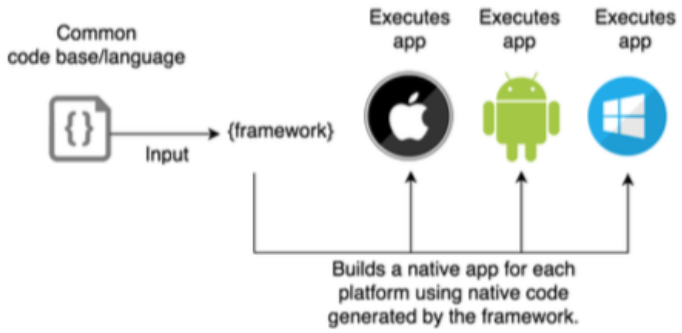


Fig. 3.5 Overview of cross-compiled build framework [16]

The apps are written in the software language Dart. Dart is object oriented with a syntax similar to java and JavaScript. The user interface for an app built with Flutter is rendered as a native interface component. The Flutter framework provides unit tests, widget tests and integrating tests for testing the app during the development process.

Installing Flutter

Flutter is installed from [4]. Installing Flutter was easier than Ionic, and the pages were more detailed and the messages during installation were clear.

When you open a new project in Flutter, an app is created with a simple functionality. This gives you an opportunity to see how the folder structures of a project are and how an app looks at an Android emulator or an iPhone simulator. One thing that made Flutter unclear

was that all code for the example apps were written in one main-file. Also, some of the tutorials had it the same way. However, at some point the tutorial started to split the code into separate files. This is something that was confusing and you easily get a feeling of losing control of your app since you are in a learning phase.

In Flutter (see Figure 3.6) to see the different layouts for the apps you open an iOS simulator or an Android emulator. In Flutter you can not have both platforms updated at the same time.

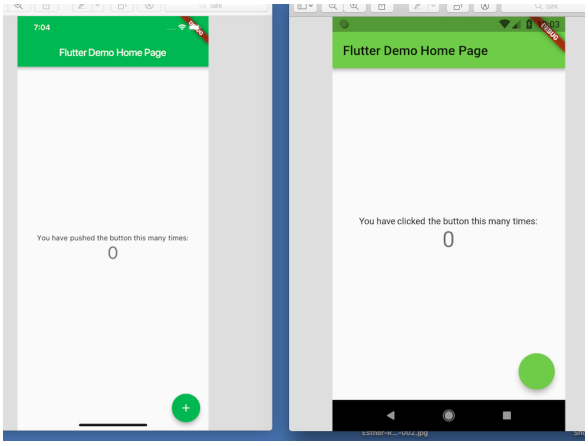


Fig. 3.6 iOS simulator and Android emulator for a template app of Flutter

Just to get some experience one may try out creating a new page and pass data between them. For Flutter it was easy to find information on how to do that.

Flutter offers a test package with the framework to write unit tests. Flutter also has solutions for creating widget tests. For the template app you find a test-folder with a test file which shows some simple tests for the app.

Flutter uses hot reload which means that when you update your source files the running Dart Virtual Machine (VM) takes the changes and updates classes with the new version of fields and functions. Then the Flutter Framework automatically rebuilds the widget tree, so that you can see the change in the emulator or simulator right away.

Conclusion of framework for developing Appidukcs

Given the experience we had with Ionic and with our background we decided that given the timeline for developing AppiDucks, Flutter was the right framework for the project. We have a background with java and the framework felt more intuitive to use. An app developed with Flutter will run on both iPhone and phones with the Android operating system, which covers the vast majority of all sold smartphones worldwide in the third quarter of 2018 [29].

3.3 Testing During Software Development

Software needs to be tested frequently during development to catch bugs, flaws and errors as early as possible. This is important since failures discovered late in a development process of a software application often become expensive and hard to fix. The goal for a test is to give the developer an objective feedback on the quality of the code and if the application meets the requirements from the customer.

In software testing you take a component and evaluate it to see if its properties respond correctly for all types of input. Testing gives the developer an opportunity to check that the software gives correct feedback for all kinds of input, and that its response time is reasonable through the whole development process. For the stakeholders, conducted tests give an understanding about how the software meets the requirements for the product.

Testing of the software can be done as soon as the software has some functionality. When to start testing of the software depends on the development process of the development team. In a more traditional approach the tests are done after much of the code is implemented. With an agile approach in the development process the tests are written and done along with the implementation of the software. Testing of software includes more than just examination of the code quality. It also examines how the code is running in different environments and under different conditions. It also gives an approval that the software is doing what it is supposed to do.

3.3.1 Testing

In software testing, a software component or system component is evaluated to see if the software meets the design and development requirements in the documentation from the customer. This is done to examine that the application responds correctly to all kinds of inputs, and that the software's function executions finish in acceptable time. Testing also checks that it is possible to install and run the software in the environment it was meant for, and finally to see that it meets the stakeholders' desires.

Even for the smallest functionality, there is an infinity of test cases you can create. Therefore, the developer has to choose what to test according to time and resources. Testing of the software gives the developer objective information of the software and how it will meet the requirements of the stakeholders.

With a *unit test* you test the smallest parts of an application, such as a function. A unit test runs a test on an independent part of the code base to see if it acts acceptably. It is the developer of the software who also writes the unit test that makes sure that it meets its design requirement and behaves as expected. Each of the test cases are independent of each other

even if they are testing the same functionality. With unit tests the developer gets confidence that each of the individual parts of the program work properly, for example testing that a function returns the expected value for a given set of inputs.

For mobile application testing in particular, you find two kinds of testing that can take place on the mobile devices. The first is hardware testing such as testing that Bluetooth, internal processors, internal hardware, camera, screen size etc. works with the application as expected. The second is testing the actual software of the application, where the software functionality of the application is tested.

The development of software for a mobile app is usually done on a PC. On the PC an emulator or simulator are used to verify general functionality and perform regression testing. Regression tests are tests that confirms that a change in the software does not impact the existing functionality of the product, like adding new functionality or changing existing features or fixing bugs.

One weakness of testing with emulators and simulators is that your app can look fine on them, but still break on some of the most popular devices. Testing mobile applications is also challenging because of all the various mobile devices that exist, like HTC, Samsung, iPhone, Huawei, etc. with their different operating systems and versions of them. They have different screen sizes, hardware configurations and virtual keypads. Together with frequent updates, this makes it harder to test a mobile application properly.

3.3.2 Test driven development

Test driven development (TDD) is a software development process which encourages simple design. It relies on the repetition of very short development cycles [34]. In TDD the work is often split into tasks. A task is a subset of the requirements for the application which can be implemented in a few days or less. The requirements for the software are turned into small test cases, and the software is improved to pass the new test. In TDD, adding a new feature to the code base always start with writing the test first.

1. First you write a test for the new feature, this small test will fail since you have not yet written the code for the functionality the test is testing. This is done to make sure that the test can fail.

2. You improve the code of the program, by implementing the functionality you are testing, but you only add enough code for the test to pass.

3. Finally, you refactor your code if possible. Refactoring the code means you improve the internal structure of an existing program's source code, without changing the

external behaviour of the program. After the refactoring is done, you have to run all the tests again to see that they all pass. If some tests fail, you have to fix the code before you add any new tests and functionality.

This process is iterated, with new tests for every new feature you want to implement. For each successful iteration, the new code and the test code has to be integrated into the existing code base. When a new functionality is added to the code base its unit test and all the other unit tests in the code base have to run without errors. Otherwise the new functionality is not considered properly implemented. In [28] they point out that one of the benefits of TDD is that the code development is kept within the developer's intellectual control since she or he is continuously making small design and implementation decisions and increasing functionality at a relatively consistent rate. This is increasing the developers' confidence that the code is working as it supposed to do.

Chapter 4

Developing AppiDucks

In this chapter we describe the implementation of the requirements for Appiducks, making a prototype with 3 week's of questions from the syllabus of INF100. The code is available at [5]

4.1 Widgets in Flutter

Flutter is a framework to create native looking apps for Android and iOS with one code base [4]. The building blocks for apps built with Flutter are the widgets. A widget is an immutable declaration of a part of the user interface (UI). A widget describes what its view should look like given a current configuration and state. A widget can define a structural element (like a button), stylistic elements (like fonts) or aspects of the layout (like columns and padding), and so on. The widgets create a hierarchy based on composition, where widgets are nested inside each other, and inherit properties from their parents. The UI for Flutter apps is built out of combining many different simple widgets that creates a widget tree for the UI (see Figure 4.1).

When you create a UI for your app you build a new widget that combines different widgets together. The new widget can be either a StatelessWidget or a StatefulWidget. A StatelessWidget is a widget that, after it is built, will not change state. A StatefulWidget on the other hand can have many different states. A StatefulWidget generates a state object that holds and remembers the information of the widget´s state.

The toolkit that comes with Flutter uses a layout algorithm on the widget tree where the constraints such as min and max width and min and max height are passed down via the parent object calling their children. The children recursively perform their layout and return the geometry up the tree.

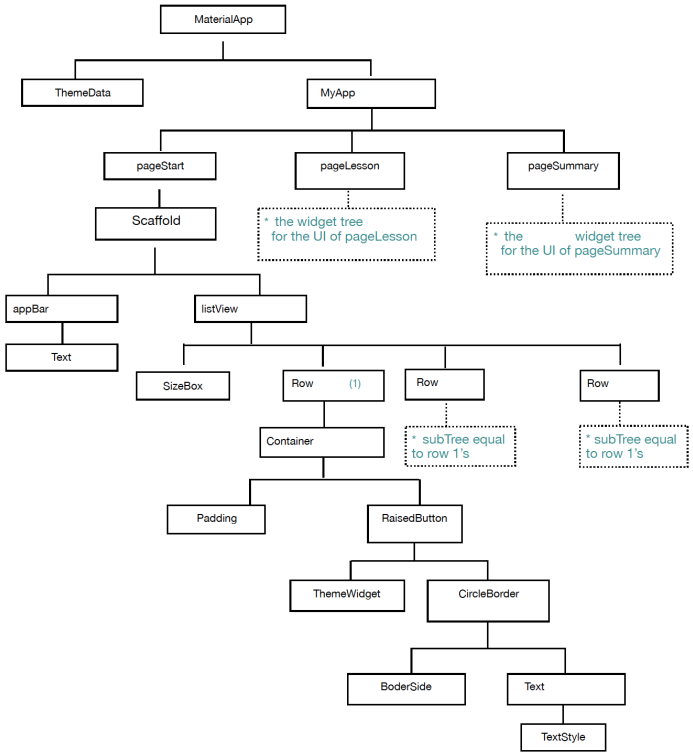


Fig. 4.1 The Widget tree for the app

When the widget has been built with the building algorithm, a related element is created with reference to the widget that created it. The element is put in the element tree that retains the logical structure of the UI. For a StatefulWidget the element also holds the reference state objects the widget created. The element tree is needed in Flutter apps since it holds the elements of what is actually on the screen, while the widget tree is a blue print of what the elements are made from.

When a StatefulWidget changes it gets a new state. The Flutter framework compares the new state with the old state of the widget, to determine the minimal changes needed of the underlying render tree to go from one state to the next. The framework does this by keeping a list of the elements that have changed and goes directly to them during the rebuild phase and skips the elements that have not changed.

4.2 Different widgets created in Appiducks

To create the three pages of Appiducks, we created two StatelessWidgets, the startPage and the summaryPage, and one StatefulWidget, the lessonPage.

All the widgets' UI for the three pages have built their UI composing different basic widgets that come with Flutter. They all use the scaffold widget to create the basic material design layout and the listView to make the pages' scroll bar and that the user can use the app both horizontally and vertically. The use of the basic material design layout gets the app to look the same on both iOS and Android platform without customization

In addition, at the three pages we also used widgets like padding-, row- and container-widgets to help arrange where we want the elements, like buttons when they are displayed on the screen.

Both the start page and the summary page use only the basic widgets that come with Flutter for the visible layout. The widget startPage has three raised buttons, for the three weeks with questions. The summary page has a textWidget for text, the container widget that draws the image, and a materialButton widget for the button to start over.

The StatefulWidget lessonPage is the page that is displayed on the screen through a lesson with 12 questions. The lessonPage widget has a basic materialButton for the next-button on the page. It also has a method currentView that returns the widget that builds up the layout for the question the user is about to answer. When the user clicks on the next-button, the method screenUptdate is called and setState changes the state for the lessonPage widget. The widgets have then changed state and have to be rebuilt and the next question's layout is shown on the lessonPage.

4.2.1 Layout for the different questions

We have created 10 different widgets for covering the layout for all the questions. These widgets' layouts are combinations of layouts for the 3 main layouts for the question types A, B and C. Type A has a question and 4 buttons with answer alternatives. Type B has a question and a text field where the user types in the answer and a button to check the given answer. Type C has an expression to evaluate, six buttons with answer alternatives, one button where the chosen answer shows up and a button to check the given answer.

Implementation of question type A

For the implementation of questions of type A, we created a StatetfulWidget called QuestionTypeA. This widget holds the UI-components Text with the question, and four buttons with the four answer alternatives, (see Figure 4.2).

The user clicks on one of the buttons to give an answer, and the user gets a pop-up message with feedback.

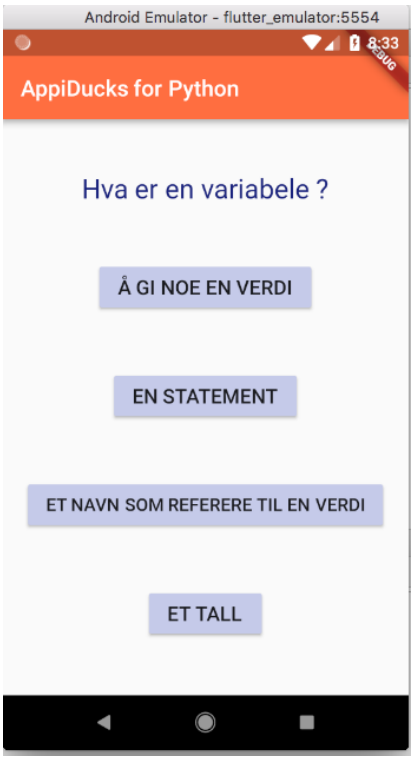


Fig. 4.2 The screen for QuestionTypeA

Implementation of the layout for question type B

For the implementation of question type B, we created a StatefulWidget called QuestionTypeB. This widget holds the UI-components Text with the question, a text-field where the user can type in the answer, and a button to click to check the answer, see Figure 4.3. For QuestionTypeB the user will also get a pop-up message like for QuestionTypeA. In addition a pop-up message will show up on the screen, if the user tries to check the answer without having typed any answer yet, see Figure 4.3b.

Implementation of the layout for question type C

For the implementation of question type C, we created a StatefulWidget called QuestionTypeC. The widget holds the UI-components Text with the question, a small button on the side, and six buttons with answer alternatives on them, see figure 4.4. The user clicks on one of the buttons and the answer shows up on the button next to the question. The user then clicks the check button to check the answer.

(a) UI for question type B.

(b) Message when trying to check the answer when no answer has been given.

Fig. 4.3 The screen for QuestionTypeB

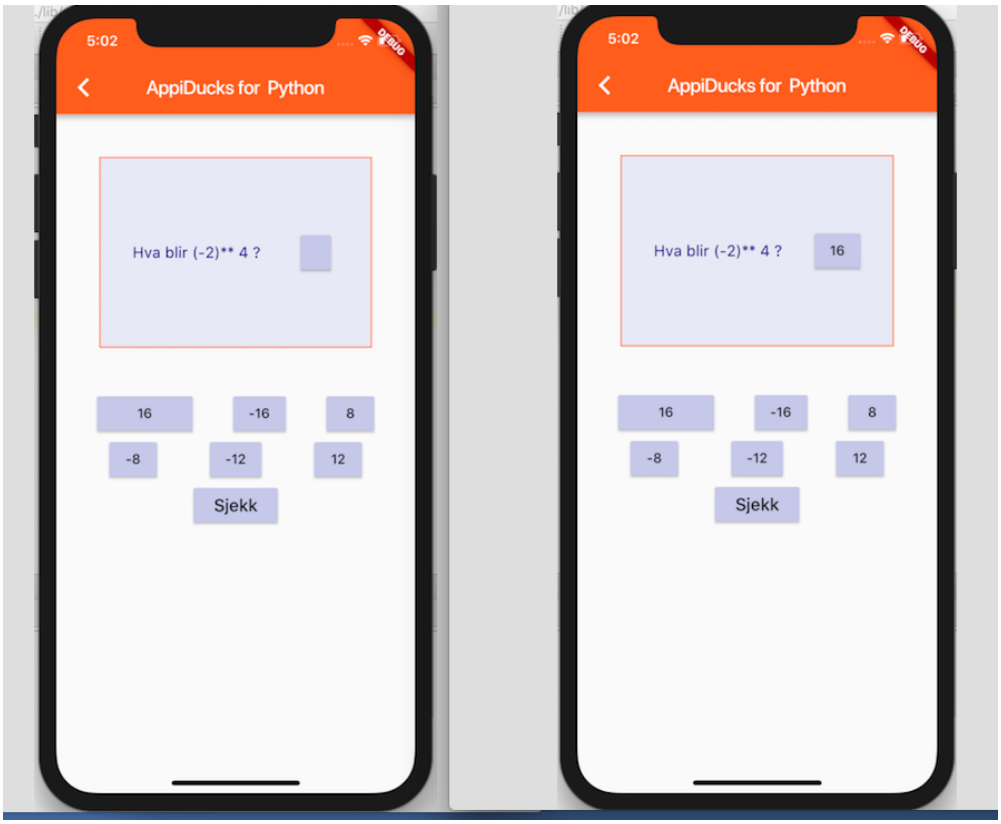


Fig. 4.4 The screen for QuestionTypeC

4.3 Implementation of questions in the app

The app is built up of 3 weeks with questions. A lesson will contain a subset of the questions from the week the user has chosen.

The questions in the app are implemented in two ways. First there are the general questions where the question, answer alternatives and the correct answer are read from a text-file. The other question types are created as methods where we use a template for the part that is constant for each question. The parts that can vary in the questions are randomly generated from a list of strings.

4.3.1 Implementation of general questions

A week's general questions are saved in a text file. When the user chooses a week, that week's general questions file is read, and question objects are created and put in a list that holds all the general questions for a lesson. When the lesson class' createLesson-method is called, it takes the generalQuestion list's length and chooses random questions from that list. The general question has layout of type A.

4.3.2 Implementation of questions in week 1

The requirements for the questions are found in Section 2.4.1.

1. Math questions

A math question has layout of type B and is of the following form:

```
Hva blir verdien til x :
    x = X  op  Y
```

where X and Y are two random integers and op is an operator randomly chosen from an operator-list which contain the following 6 operators : *,**, //,/, floor and modulo. It has no answer alternatives, the user types in her answer.

2. precedence questions

A precedence question has layout of type C and has one of these three forms :

```
Hva er resultatet av :  ( X op1 Y ) op2 Z ?

Hva er resultatet av :  X op1 ( Y op2  Z ) ?

Hva er resultatet av :  X op1  Y op2   Z  ?
```

Where X, Y and Z are three random integers. The operators op1 and op2 are randomly chosen from an operator-list which contain the following 5 operators: + , - , * , // , / . The five wrong answer alternatives are made out of the sum of the correct answer and a randomly chosen number between -5 and 4.

3. Variables and assignment questions

An assignment question has layout of type A and takes one of these two forms:

```
Hva blir x = Number1 op Number2 ?
Hva blir x = Number1 ?
```

Here Number1 and Number2 are both random integers or random doubles. The op is randomly chosen from an operator-list which contain the following 5 operators:'no operator',+,*,-,**. The three wrong answer alternatives are made out of the sum of the correct answer and a randomly chosen number between -5 and 4

4. Type questions

The type question has layout of type A and is of the form :

```
Hvilken type er Number1 ?
```

The number Number1 has a type randomly chosen from the two types Integer or Float. Depending on the randomly chosen type, Number1 is either a random int or double number. The four answer alternatives are the two types Integer and Float, together with a randomly chosen Number1 and a randomly chosen number2.

5. Augmented assignment questions

The Augmented assignment questions has layout of type A and has the form

```
Hva blir
x =  Number1
x op = Number2
```

where Number1 is a random integer. The operator op is randomly chosen from an operator-list which contain the following 6 operators: +=,-=,*=,**,/=,//= . The three answer alternatives are made out of the sum of the correct answer and a randomly chosen number between -5 and 4.

4.3.3 Implementation of questions in week 2

The requirements for these questions are found in Section 2.4.2

1. string length

The string length question has the layout of type B and has the form

```
Hva blir resultatet av kallet:
len("Word1") ?
```

Word1 is a randomly chosen string form a list of 68 different strings with different lengths. The user types in the answer.

Concatenation

The concatenation questions are expressions of two or three strings or integer numbers combined together. The correct answer for the question, is what the result the concatenation expression will return. As answer alternatives, three other wrong results are created for the concatenation of the expression.

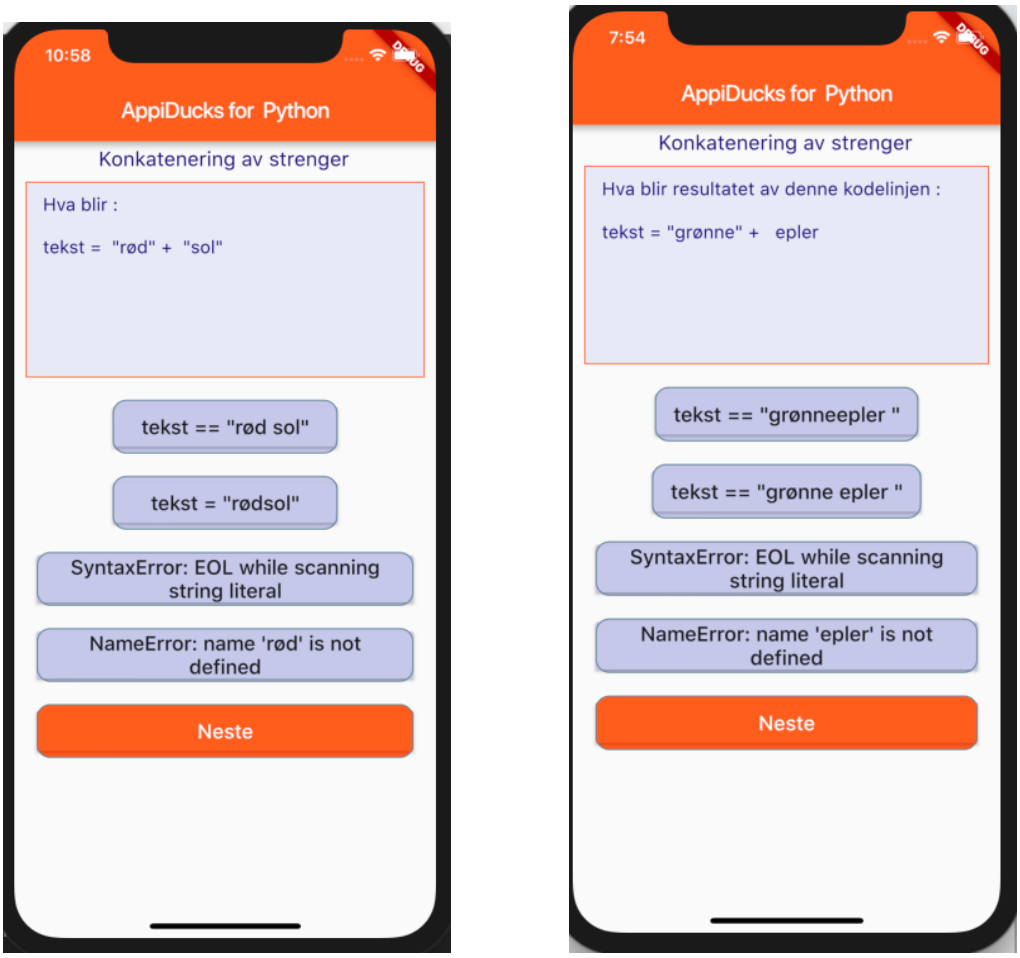
2. Concatenation with two strings

The question is 'Hva blir resultatet av denne kodelinjen: expression'. For the concatenation questions with two strings we have made a list of eight different results you can get when you try to construct an expression with concatenations. The following list shows how the expression will appear on the screen for all the eight results and the correct answer.

```
(1)  tekst = "ord1"   *   "ord2"
with correct answer : "TypeError: can't multiply sequence
                                  by non-int of type 'str' "
(2)  tekst = "ord1" +  ord2
with correct answer :   "NameError: name '" + ord2 + "' is not defined"
(3)  tekst ="ord1" + "ord2"
with correct answer : tekst == "ord1ord2"
(4)  tekst = " ord1 " + " ord2 "
with correct answer : tekst == " ord1 ord2 "
(5)  tekst = " ord1 "  " ord2
with correct answer : SyntaxError: EOL while scanning string literal
(6)  tekst =  ord1 + ord2
with correct anwer : SyntaxError: invalid syntax
(7)  tekst = " ord1  ord2  "
with correct answer : tekst == " ord1 ord2 "
(8) tekst =  ord1 ord2
with correct answer : SyntaxError: invalid syntax
```

See Figure 4.13 for two examples.

The variables ord1 and ord2 are randomly chosen form a list of 26 strings, where the strings are of the form "ord1 | ord2". The String is split at the |- sign and ord1 and ord2 are put in the code so the expression becomes the randomly chosen result of the concatenation. See Figure 4.5a for examples where concatenation is made from result label 'utenMellomRom', tekst = "ord1" + "ord2" and see Figure 4.5b where concatenation is made from the result label 'nameError', tekst= "word1" + word2.



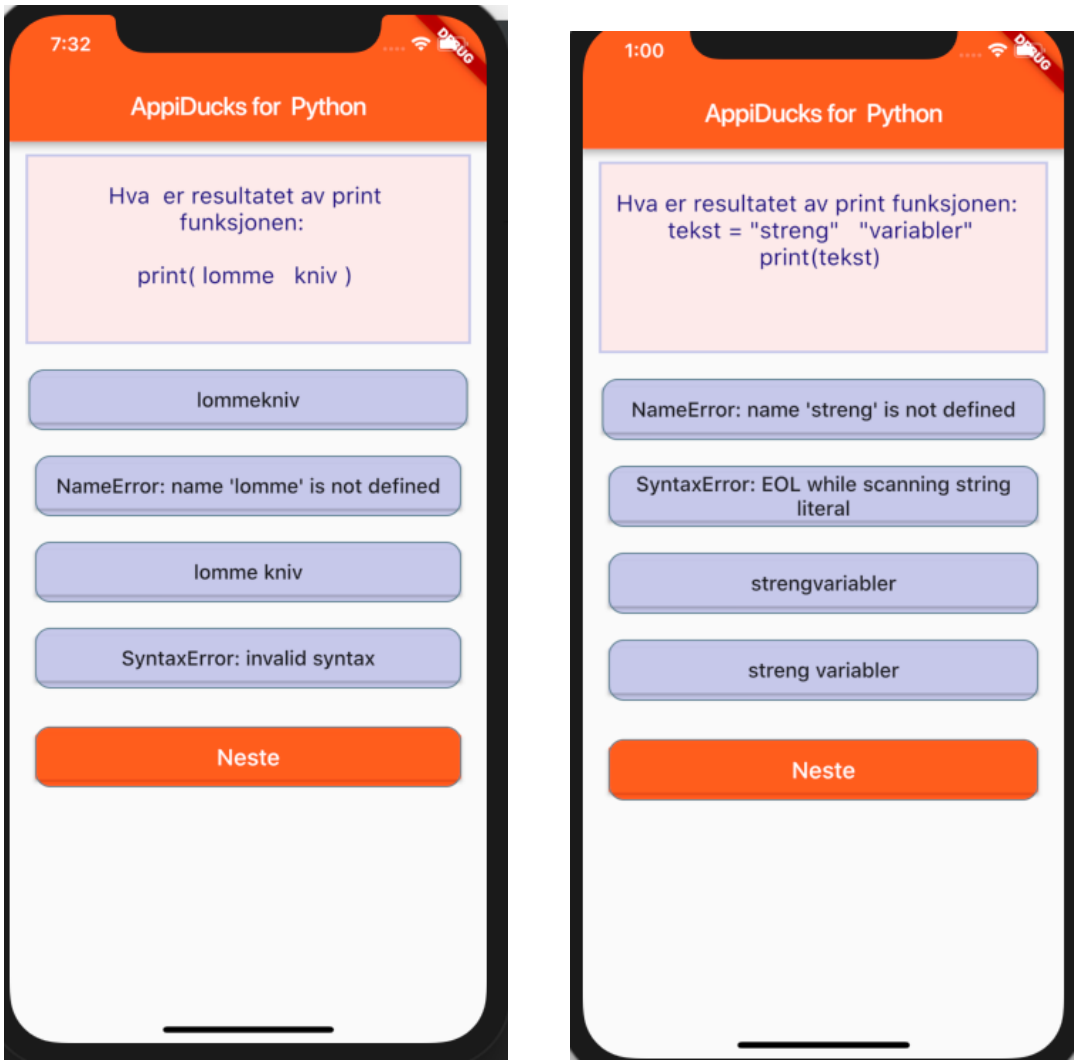
(a) Concatenation with result 'utenMellomRom (b) Concatenation with result 'nameError'

Fig. 4.5 Concatenation with two String

3. Print question

The print-questions is 'Hva er resultatet av print funksjonen : print(expression)'. The correct answer of the question is the result in Python of executing the print expression. The expression contains strings, integers or integers and words combined together, see Figure 4.6. We have made 7 different results the print-question can have. Below is the print-function with the expression and how the two ways will show on the screen, and the 7 results that forms the correct answer.

```
(1)    tekst = "ord1" +  ord2 "                    print("ord1" +  ord2 ")
       print(tekst)

       with correct answer :  SyntaxError: EOL while scanning string literal


(2)    tekst = "ord1" + "ord2"                       print("ord1"+"ord2")
       print(tekst)

       with correct answer : ord1ord2


(3)    tekst = "ord1"   "ord2"          or       print("ord1"   "ord2")
       print(tekst)
       with correct answer : ord1ord2

(4)    print( ord1  ord2 )
       with correct answer : SyntaxError: invalid syntax

(5)   tekst = " ord1 " + " ord2 "       or       print("ord1 " + " ord2")
      print(tekst)
   with correct answer : ord1 ord2

(6)  tekst = " ord1  ord2              or       print("ord1  ord2)
     print(tekst)');
     with correct answer : SyntaxError: EOL while scanning string literal

(7)   tekst = "ord1  ord2 "
```



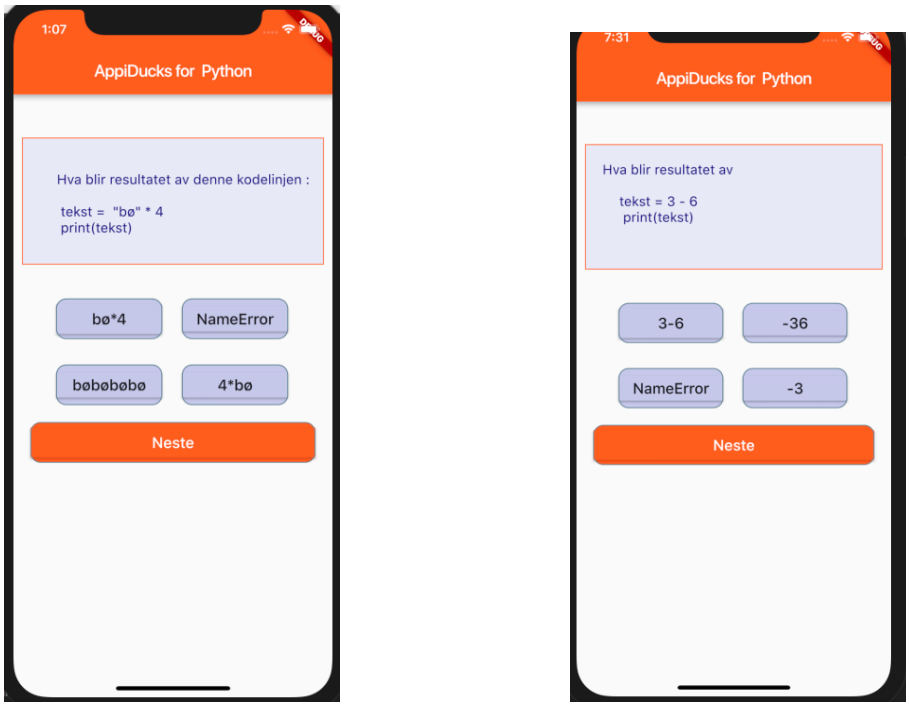
(a) print(ord1 ord2) (b) Print question with strings

Fig. 4.6 Print questions

```
print("tekst")
with correct answer : tekst
```

The ord1 and ord2 are split from a string of the form ord1|ord2. The string is randomly chosen form a list of 26 strings. The string is split at the |-sign and then put together in a way that gives a randomly chosen result-list for the print-function, including the correct answer. See Figure 4.6 for an example of a print question with strings.

For the print-function questions with random integers, the integers can be treated like strings that are printed, or like two integers that are calculated together with an operator randomly chosen from a list of these four operators : '+', '*', '-', '/', see Figure 4.7b.



(a) print(word*integer) (b) print question with calculation of two integers

Fig. 4.7 print function

The print-function in Python also handles word*Integer, see Figure 4.7a. The word is randomly chosen from a list of seven strings. The string*integer we have implemented as concatenating the word in a for-loop.

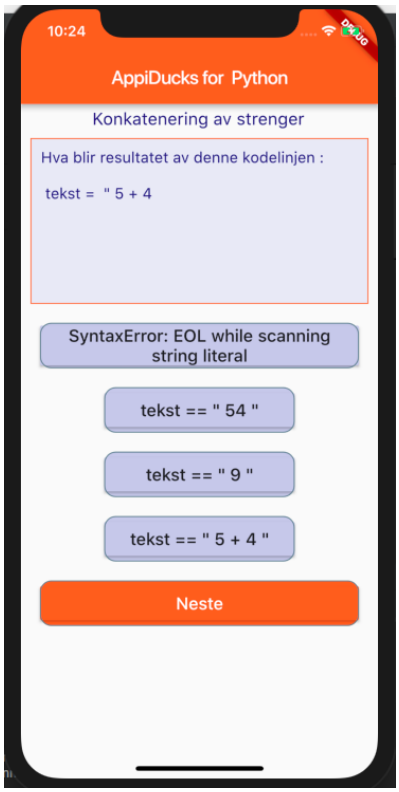
4. Concatenation with two integers

For the concatenation questions with random integers, the integers can be treated like strings that are concatenated, or like two integers that are calculated together. For the questions with a calculation, the operator is randomly chosen from a list with these four operators : '+', '*', '-', '/'. In addition, the concatenation can be an expression on the form word*integer, where word is randomly chosen from a list of seven strings, see Figure 4.8.

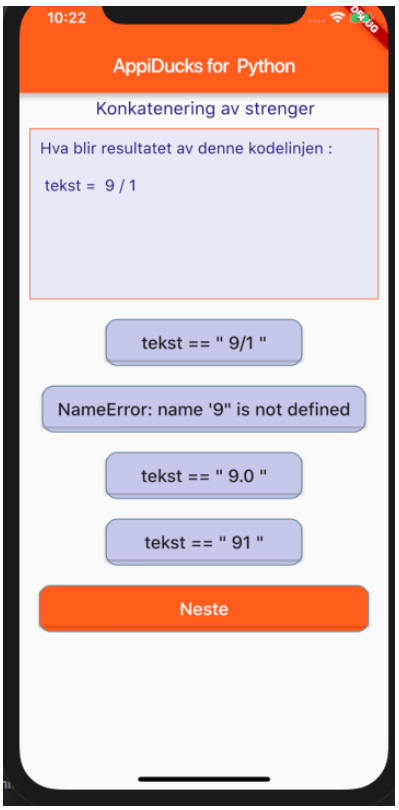
5. Concatenation questions with three strings

The question is 'Hva blir resultatet av denne kodelinjen: expression'. For the concatenation questions with three strings we have used the same list with 8 results that is used for the concatenation question with two strings.

```
(1)  tekst = "ord1"   +   "ord2"  * "ord3"
```



(a) result label 'syntaxError' tekst= "num1 + num2



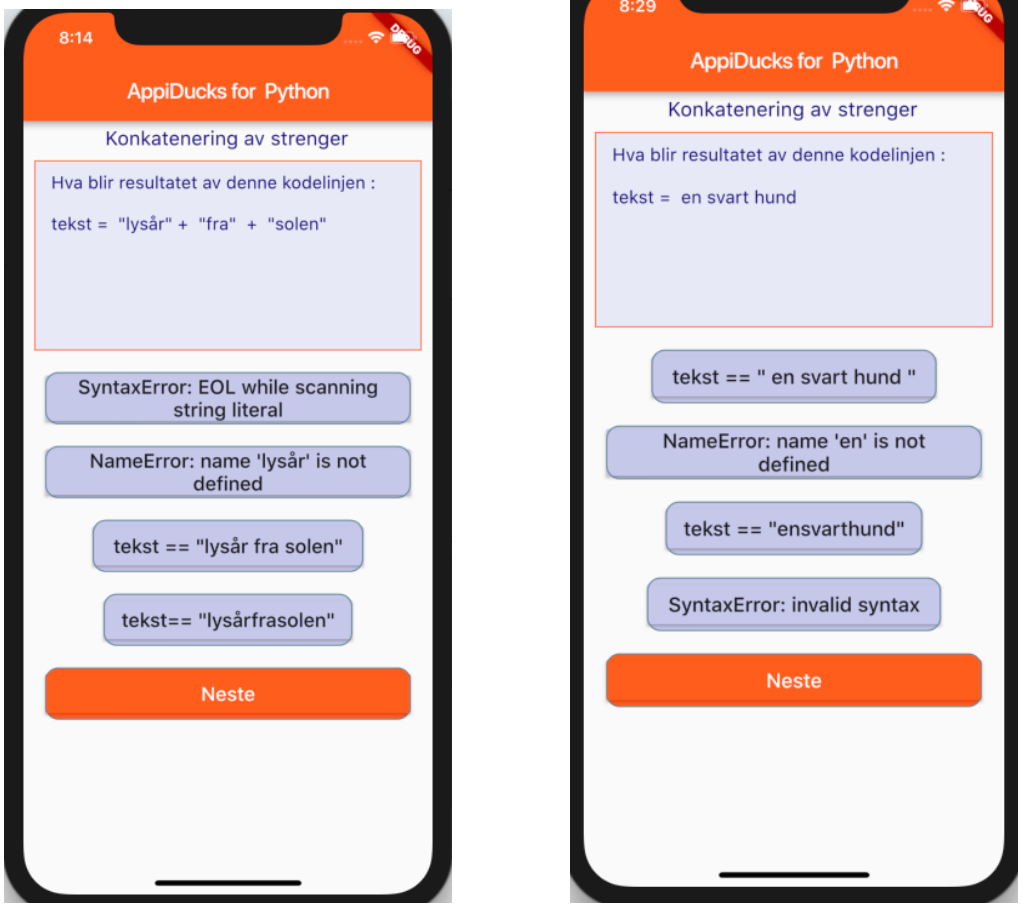
(b) result label 'beregne' tekst= num1 / num2

Fig. 4.8 print with integer

```
with correct answer : "TypeError: can't multiply sequence
                                  by non-int of type 'str' "
(2)  tekst = "ord1" +  ord2  + ord3
with coorect answer :   "NameError: name '" + ord2 + "' is not defined"
(3)  tekst ="ord1" + "ord2" + "ord3"
with correct answer :tekst == "ord1ord2ord3"
.
.
.
(8) tekst =  ord1 ord2 ord3
with correct answer : SyntaxError: invalid syntax
```

The strings ord1, ord2 and ord3 are randomly chosen from a list of 18 strings, where the strings are on the form "ord1|ord2|ord3". The string is split at the |- sign and ord1, ord2 and ord3 are put together to make an expression that gives the randomly chosen result of the concatenation. See Figure 4.9a, where the concatenation is made from the result label

'utenMellomRom', tekst = "ord1" + "ord2" +"ord3". In Figure 4.9b the concatenation is made from result label 'syntaxError', tekst= ord1 word2 ord3



(a) result label 'utenMellomRom', tekst = "ord1" + "ord2" +"ord3"

(b) result label 'syntaxError', tekst= ord1 word2 ord3

4.3.4 Implementation of questions in week 3

The requirement for the questions are found in Section 2.4.3.

1. 'built-in functions' question

The built-in function question has layout of type B and the form:

```
Hva returnerer den innebygde funksjonen
        functionName()
```

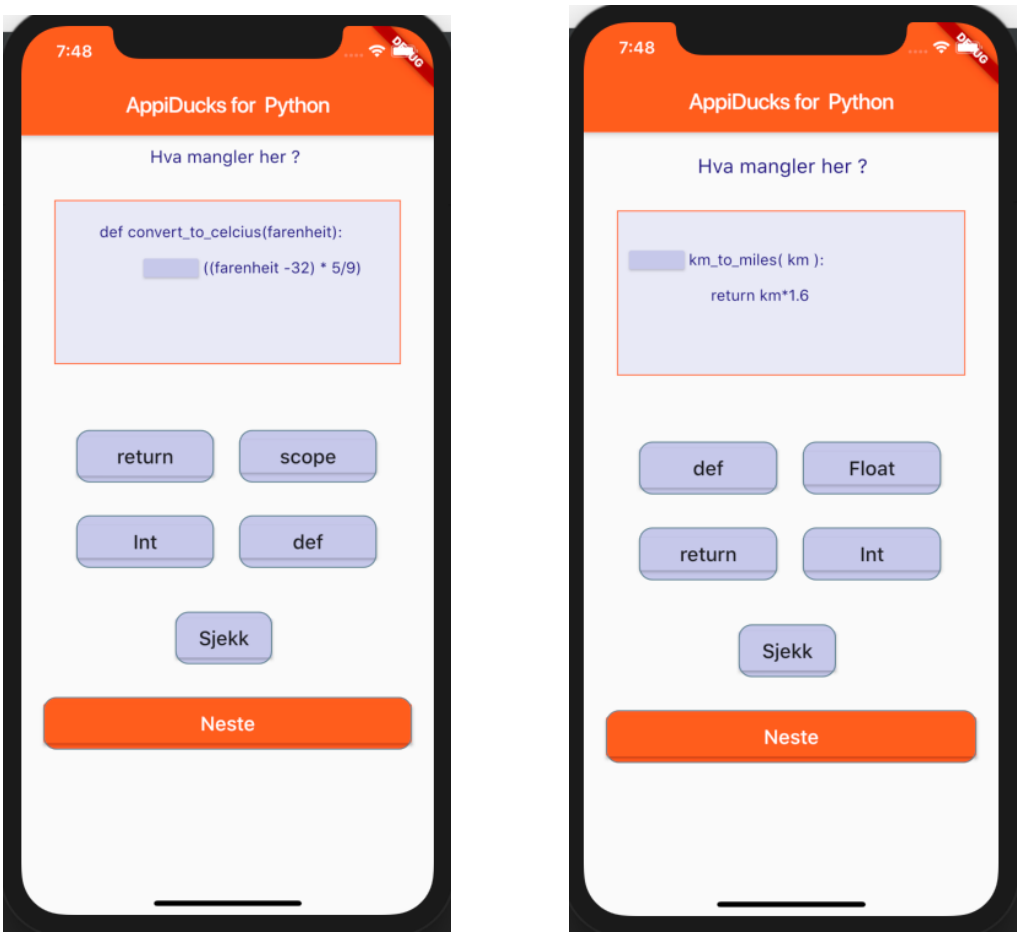
The functionName is randomly chosen from a list of strings naming 4 built-in functions that come with Python.

2. 'missing code' part questions

The question is' Hva mangler her ?' with a small piece of code that has a missing part, see Figure 4.10. What part of code that is missing, def or return, is randomly chosen. The small code that the question can contain have been made from 4 different templates, see below. The functionName, operatorName and operator will be filled into Template 1 and 3 when the question is created, and for Templates 2 and 4 the fuctionName, argument and returUtrykk are filled in.

```
          Hva mangler her ?

(1)          ____ functionName( x, y):
           """ Return x operatorName y """
              return x operator y

(2)         Hva mangler her ?
           ____ functionName(argument):
               return returUtrykk

(3)          Hva mangler her ?
          def functionName( x, y):
           """ Return x operatorName y """
              _____ x operator y

(4)         Hva mangler her ?
          def functionName(argument):
              ______ returUtrykk
```

The parts in the code that are filled in are randomly chosen from a list of 12 strings that have the form: "**functionName|argument|returUtrykk|operatorName|operator**". The string is split at the |-signs and the different parts of the string are filled into the template of the code. See Figure 4.10a where 'return' is the missing part of the code. The function that is shown in the code is created from the string "add|""|""|pluss|+" with Template 3.



(a) Missing code question where return is missing (b) Missing code question where def is missing

Fig. 4.10 Missing code part

3. 'What does this code return' question

The question is 'Hva vil denne koden returnere ?' with a small code part. The code in the question is made from 4 templates, where the functionName, operatorName and operator are filled in when the question is created, and where num1, num2 and num3 are random integers.

```
(1)     Hva vil denne koden returnere ?

        def functionName(x,y):
           """Return x operatorName x """
              return x operator y
        functionName(num1, num2)

(2)     Hva vil denne koden returnere ?
```

```
x = num1
y = num2
def functionName( x, y ):
"""Return x operatorName x """
      return x operator y
functionName(num1, num2)
```

(3)

```
Hva vil denne koden returnere ?
x = num1
y = num2
def functionName( x, y ):
"""Return x operatorName x """
      x = num3
      return x operator y
functionName(num1, num2)
```

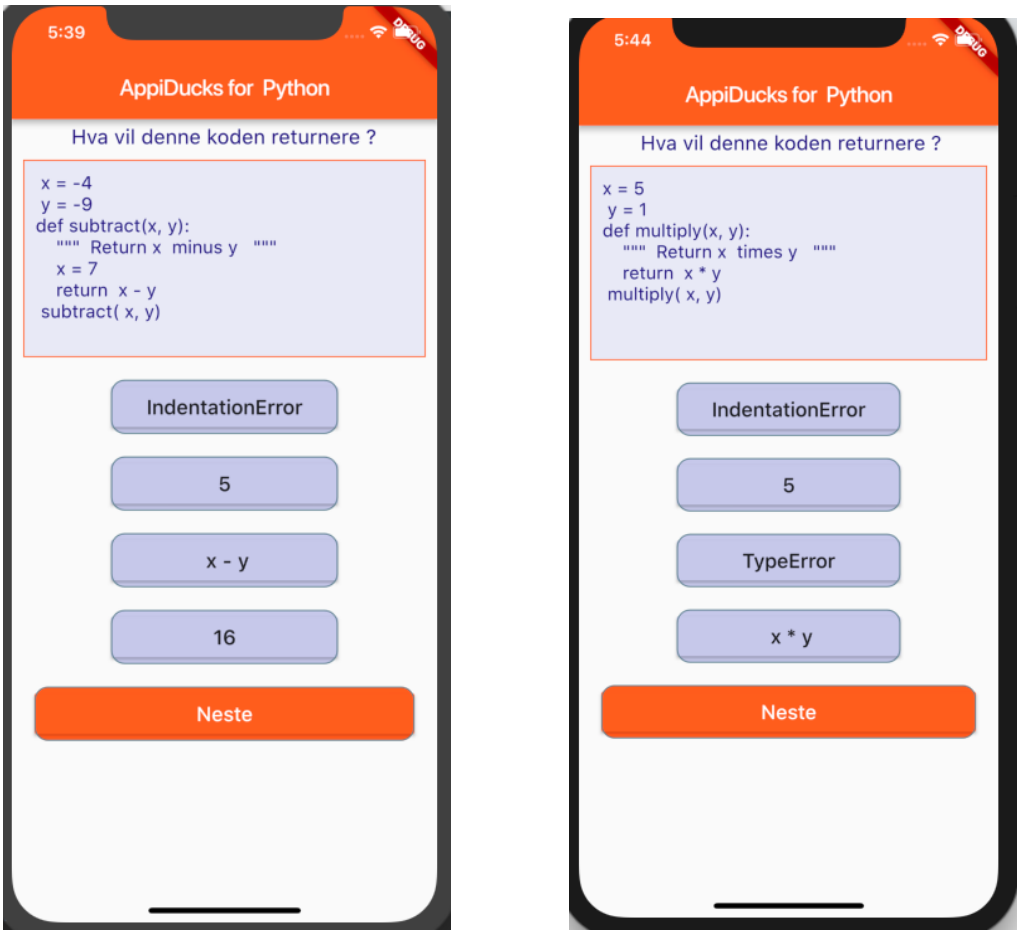
(4)

```
Hva vil denne koden returnere ?
x = num1
y = num2
def functionName( x, y ):
"""Return x operatorName x """
      y = num3
      return x operator y
functionName(num1, num2)
```

The parts in the code that are filled in are randomly chosen from a list of 19 strings that have the form: "**kodeX|functionName|operatorName|operator**". The string is split at the |-signs. The kodeX part decides which template to use. The rest of the parts in the string are filled into that template of code. See fig 4.11a, which shows an example of code created from the string "3|subtract|minus|-" with Template 3.

3. Question about error-messages

The question is 'Hva vil denne koden returnere ?' with a small code part. The code in question is made from 3 templates, where the functionName and operator are filled in when the question is created, and where num1, num2 and num3 are random integers.

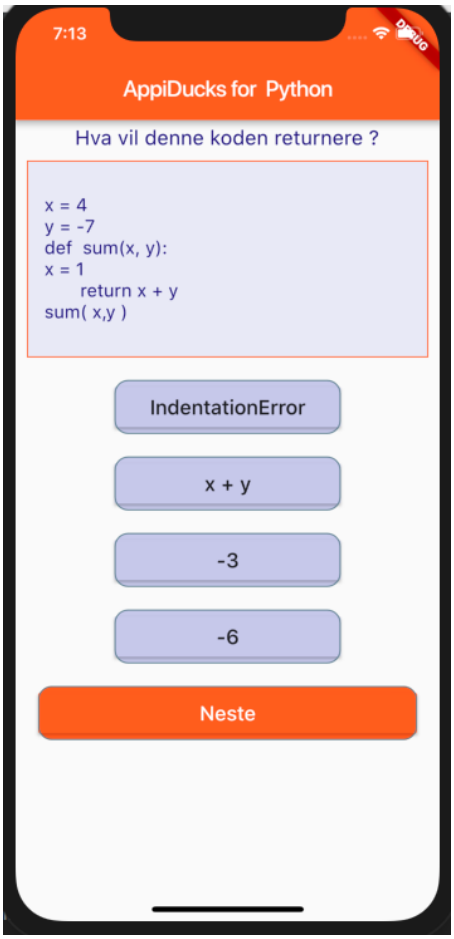


(a) Return code question made out of Template 3 (b) Return code question made out of Template 2

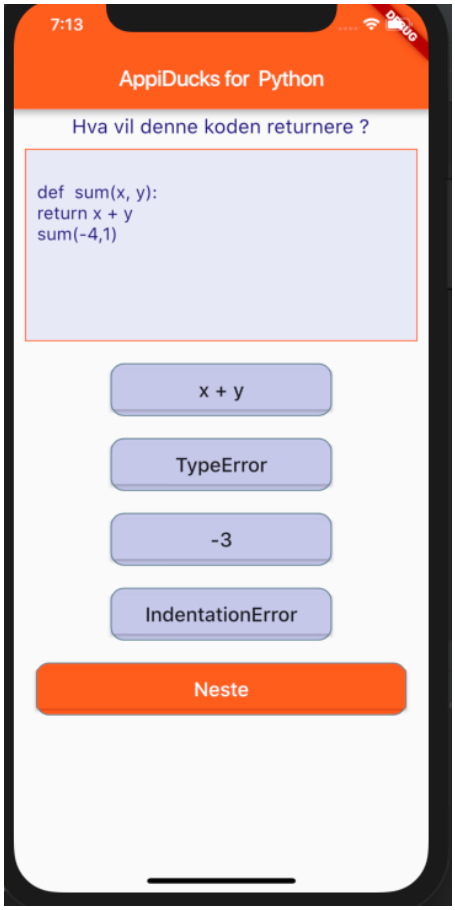
Fig. 4.11 What does the code return questions

```
(1)   Hva vil denne koden returnere ?
      def functionName(x,y):
      return x op y
      functionName(x,y)

(2)   Hva vil denne koden returnere ?
       x = num1
       y = num2
       def functionName(x, y):
       return x op y
       funtionName(x, y)

(2)    Hva vil denne koden returnere ?
```

```
x = num1
y = num2
def functionName(x, y):
x = num3
return x op y
functionName(x, y)
```



(a) Error-message question from Template 3



(b) Error-message question from Template 1

Fig. 4.12 Error message questions

The parts in the code that are filled in are randomly chosen from a list of seven strings that has the form: "**typeX|functionName|operator**". The string is split at the |-signs, and the typeX decides which template to use. The rest of the part in the string is filled into that template of code. See Figure 4.12a that shows how the code is created from the string "3|sum|pluss|-" with Template 3.

4.4 Implementation of lessons in the app

The prototype of the app has 3 weeks of questions that a lesson is created from. The requirements for which question types the lesson for that week should contain is found in Sections 2.4.1, 2.4.2 and 2.4.3. It was not possible to implement a general method that we could use for all the three lessons. For that reason we have implemented three different methods for the three lessons. We chose that solution in the implementation since the prototype only contains three weeks and it needed to be finished by the beginning of the semester for doing a user survey.

If all of the 12 weeks should have been implemented this would not have been a good solution. We have in the implementation thought of how to make a lesson feel interesting in the sense that the question layout is changing. This is done in the way that a lesson does not contain two equal question types in a row.

4.5 Development Experience and Evaluation

4.5.1 Our experience in developing AppiDucks

Flutter and learning apps

Building an app with the Flutter framework results in an app built with one code base available for both Android and iOS. The finished prototype of the app looks and acts the same on both types of phones.

Flutter had been released only one month before we started developing AppiDucks, as Preview1 in July 2018. For that reason the information about the framework was a bit immature. This was something that made it hard in the beginning of the development process. It was difficult to find info and tutorials about how to do things. For instance, when we implemented the UI for the different questions the widget tree became very big (see Figure 4.1).

Was it right, or was there a better way to do it? Should we have split them into smaller parts?

In the development process some small differences appeared. One came when implementing QuestionTypeB. The bottom of the screen overflowed by 103 pixels for the Android emulator while the iOS simulator was fine, see Figure 4.13. The solution for this was simply to use the ListView-widget that comes with Flutter for the view of the screen.

Some differences also appeared between the two operating systems when we tried to implement a number keyboard for QuestionTypeB, see Figure 4.14.

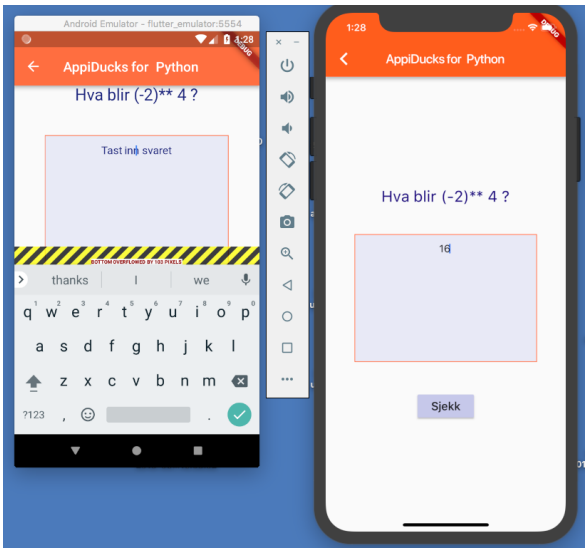


Fig. 4.13 Difference in the layout in iOS simulator (to the right) and Android emulator (to the left), question type B

The keyboard is needed for the user to type in their answer. On the Android emulator the number keyboard popped up, but for the iPhone simulator there was no keyboard that popped up. We chose to solve this by just using the regular keyboard for the device.

We also experienced differences between the iPhone simulator and an actual phone, where the text went outside the screen on the phone, but not in the iPhone simulator version for the same phone. We fixed this by adjusting the size of the content so that it also fit the size of the physical phone screen.

In the prototype, no built-in functionality that comes with the different devices, like camera or GPS etc. were used. This may be a reason for why there were not more differences between the two operating systems when we were developing the app.

All the different screen sizes and versions of the devices always make it difficult to know how the finished app will look on a real device. But overall, we found Flutter to be easy to understand and develop an app with.

After developing a prototype of a learning app as a supplement to a course our thought is that developing an app demands a lot of technology knowledge. This requires skills a teacher will not normally have. It will be hard to expect that an ordinary teacher should acquire these skills for making a learning app for their classes.

It will still be important to have the teacher as part of the development process in an early state. The teacher needs to be heard about what kind of question types, interactions, and layout the app should have. This is to make a learning app that has relevant content, tasks

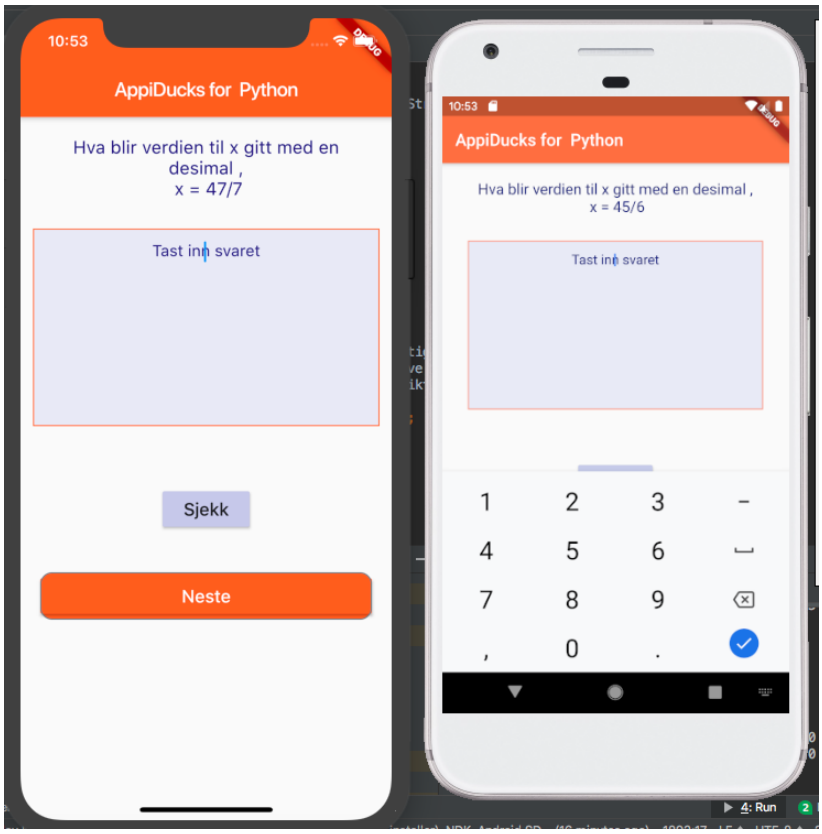


Fig. 4.14 Number keyboard not popping up on iOS simulator (to the left), but appearing in the Android emulator (to the right), question type B

and level for the subject the app is made for. What content to have and how it should be presented is something that can vary between different topics.

The finished app should therefore be developed as a framework used as a template so it is easy for the education institute to reuse the app among several courses. This means that the teacher of the course can be responsible for the content of the learning app for the classes they teach. Creating content that is big enough to give an app varying questions and tasks will be a job on its own for the teacher. A template will also make it easier to change the content of an app if the syllabus of a course changes.

4.5.2 Testing while developing AppiDucks

The intention was to write unit tests and use a TDD approach in the development process of AppiDucks. Our experience with Flutter was that it was not so straight forward to do unit or widget tests, even though the framework comes with test packages for unit and widget tests. We therefore decided that with the time we had available for this relatively small and simple

app, it would have been too time-consuming to develop separated unit and widget tests in the development process.

We therefore went from trying to write unit and widget tests to use regression tests of the app under its development. The hot reload functionality that comes with the Flutter framework made it easy. With the hot reload you see changes you have made immediately on the emulators and simulators that come with the framework, or even on a physical phone. Especially when the changes were done at the app's UI the hot reload was very useful. The change in the UI appeared directly at the emulator with no interruption in the developing process.

When we had implemented a new question type, we made a test-lesson that only contained 12 questions of the new question type and manually checked a couple of times that the question appeared correctly on the screen and had the correct answer and answer alternatives to it.

When AppiDucks was ready for the user survey it was tested manually on the following types of devices to verify how the app looked. The first four devices have an Android operating system and the last one has iOS.

- Huawei, Honor 8 lite
- Samsung 8
- Sony Xperia xz2
- Huawei, P smart
- iPhone 6

On all devices the app was fine. There were no problems with opening it and clicking through lessons. The feedback we got from the people that tested the app was more about the layout in the app.

One issue was if it was possible to get the whole content to fit on the screen without having to scroll to see it. The screen size of the different devices is a common problem in app development because of all the different types of devices on market.

We also got some comment on the color of the feedback message the user gets when answering a question. The feedback we got was that for the right answer the color should be green, maybe only with a symbol, and that the feedback for wrong answer should be in red. We have chosen to have it in blue because it is the complementary color to orange which is the main color in the app. By using those two colors, the app is given a color harmony.

But we also got some feedback that the app's lessons should have had a progress bar that shows how much the user has done of a lesson when answering the questions. We had unfortunately not time to implement this before we started the user survey.

Chapter 5

User survey of the app

In this chapter we describe the execution of a user survey among INF100 students and discuss its results.

A prototype of the app was implemented with questions from 3 weeks. The app was uploaded to a server to be available for the user survey, and the participants were students that took INF100. The user survey was done in the beginning of the spring semester in 2019.

The participants in the survey were in two groups, one that used the app for three weeks and a control group which did not use it. Both groups took a pre-test before the three weeks started and an after-test when the three weeks were done. The results from the pre- and after-tests for the two groups were compared to see if there were any differences between those who used the app and the control group.

To make an app available for iPhone is more expensive than for Android phones. You need to get an Apple developer ID to sign the app, which costs 99 dollars for a year. For that reason we chose to let students with Android phones use the app, and students with iPhone to be in the control group for this small user survey.

5.1 Questionnaire design

Both of the tests were designed as a multiple choice questionnaire. Each question had four answer alternatives, where only one of them was the correct answer. In addition to the test questions we asked for the e-mail addresses to link the two tests a student took and compare the results of the pre-test with the results of the after-test for each student.

5.1.1 Pre-test

The pre-test contained 18 questions from the app to measure how much of the app's contents the students were familiar with before the survey began. The app's three weeks of content is from chapter 1, 2, 3, 4, and 15 of the textbook, "Practical Programming - An Introduction to Computer Science Using Python, Third Edition" by Paul Gries, Jennier Campbell and Jason Montojo.

First, the pre-test had five questions about keywords from all three weeks. The questions about keywords in the test cover basic math expressions and definitions. Next follows five precedence questions from week 1, to see if the students are familiar with the operators precedence and how the use of parentheses can change the order the expression is interpreted. The understanding of operators precedence is important when it comes to programming code that does correct calculations.

Then there are three concatenation questions from week 2 where the user is asked what the result of the concatenations of some strings are. There are two questions with two strings and one with three strings.

There is one question about what the result is when the print-function has two strings as arguments, from week 2.

From week 3 the pre-test had four questions with a small code part. This is to see if the student can read code and interpret what the result will be of executing that code part. This is a skill in program development that is needed in maintenance of source code and working in a development team with other developers. In the pre-test we also asked if the students knew any other programming languages from before. For the pre-test questions, see below.

The question in the pre-test

1. **Hvordan ser krøllparenteser ut?**
   a) () b) [] c) {} d) §§
2. **Hva betyr %=?**
   a) Operator assignment b) En verdi som refererer til et navn
   c) En verdi som refererer til et navn d) Forkortelse for remainder og assignment
3. **Hva er 'Whitespace'?**
   a) Python skall b) Max linjelendge i Python
   c) Alle tegn som normalt er usynlige som space og return d) #

4. **Hva er et 'scope'?**
   a) området av et program hvor en variabel er gyldig b) en mengde variabler
   c) stedet i minnet hvor en variabel er satt d) TestError

5. **Hva er resultatet av : (4 + 7) * 2 ?**
   a) 18 b) 20 c) 22 d) 21

6. **Hva er resultatet av : 12 // (4 // 2) ?**
   a) 1 b) 2 c) 6 d) 3

7. **Hva er resultatet av : 12 // 4 // 2 ?**
   a) 6 b) 3 c) 2 d) 1

8. **Hva blir resultatet av denne kodelinjen : tekst = sommer ferie‘ ?**
   a) tekst == ‘sommer ferie’ b) SyntaxError: EOL while scanning string literal
   c) tekst == ‘sommerferie’ d) SyntaxError: invalid syntax

9. **Hva blir resultatet av print funksjonen : print(‘vann’ ‘farger’)**
   a) SyntaxError: EOL while scanning string literal b) SyntaxError: invalid syntax
   c) vannfarger d) vann farger

10. **Hva vil denne koden returnere ?**

```
def subtract(x,y):
   ‘’’ Return x minus y ‘’’
       return x-y
subtract(-4,8)
```

   a) IndentationError b) x-y c) -12 d) TypeError

11. **Hva vil denne koden returnere ?**

```
x = -4
y = -3
def add(x,y):
   ‘’’ Return x pluss y ‘’’
```

```
        x =8
        return x + y
add(x,y)
```

a) -7 b) IndentationError c) x + y d) 5

12. **Hva vil denne koden returnere ?**

```
def subtract( x, y):
return x-y
subtract(-2,0)
```

a) -2 b) -2 c) TypeError d) IndentationError

13. **Hva vil denne koden returnere ?**

```
x= 5
y =1
def multiply(x,y):
‘’’ Return x times y ‘’’
    y = -7
    return x * y
multiply( x,y)
```

a) -35 b) IndentationError c) 5 d) x * y

14. **Hva er resultatet av : 35 - 10 / 5**
    a) 5 b) 10 c) 33 d) -25

15. **Hva er resultatet av : (-3) - 15 / 4**
    a) 20.0 b) -6.75 c) -4.50 d) -5.00

16. **Hva er ’floor’?**
    a) Heltall-biten til et float tall b) Modul
    c) Det største heltallet mindre eller lik input tallet d) Operator som tar to operander

17. **Hva blir resultatet av denne kodelinjen : tekst = ‘lomme ’ + ‘ kniv’**
    a) tekst ==‘lomme kniv’ b) SyntaxError: EOL while scanning string literal
    c) tekst == ‘lommekniv’ d) NameError: name ’lomme’ is not defined

18. **Hva blir resultatet av denne kodelinjen : tekst = ‘ student ‘ + ‘ + ’ + ‘ hybel’**
    a) SyntaxError: EOL while scanning string literal b) SyntaxError: invalid syntax
    c) tekst == ‘student + hybel’ d) tekst == ‘studenthybel’

5.1.2 After-test

The after-test had 16 questions, with two questions from the 8 different parts of the 3 first weeks of the syllabus of INF100. In the after-test we asked if the students had used the app, and if they had, how much. Here the students could choose between 1-3, 3-5, 5-10, 10-15 or more than 15 times.

Questions 1 and 2 in the test are general theory questions about Python and questions 3 and 4 are questions about expressions in Python. To get a different formulation for those four theoretical questions from how they are asked in the app, we used questions from a beginner quiz in Python [6] and [7].

In addition to the four theoretical questions in the after-test, there are 12 questions combining different tasks in the app. The students were asked to interpret what some Python code would return.

In questions 5 and 6, two strings are concatenated to a new string. The student is asked what the print-function will print when the len-function is used to find the length of the new string. Here the students are tested to see if they know the built-in len-function, and that space is included when counting the length of a string.

Questions 7 and 8 are two questions where the print-function takes a string that is created by concatenating strings and integers. These are two questions that show if the students know how concatenation is done with different types, and what the result is when the string is printed with the print-function.

Questions 9 and 10 are two precedence questions to see if the students are familiar with the change in the order the operators are done when parentheses are used.

In questions 11 and 12 there are two questions with a small code part that the students have to interpret. In question 11 the student has to know how to use augmented assignments to calculate what the output is. Question 12 is a piece of code that will return an error message.

Questions 13 and 14 are two concatenation tasks done with different types to see if the students know what the result is when a concatenation is done with two strings and what the result is when it is done with two integers.

The two last questions in the after-test are two simple math expressions to see if the students are familiar with them. See below for the actual questions in the test.

Questions in the after-test

1. **Hvilke av de følgende utrykkene er den riktige måten å starte en funksjon i Python ?**
   a) def myFunction() b) function myFunction():
   c) def myFunction(): d) function myFunction()

2. **Hvilken av disse funksjonene i Python vil skrive ut en melding på skjermen?**
   a) sys.out.println() b) console.writeln()
   c) writeln() d) print()

3. **Vil bruk av parenteser endre rekkefølgen Python evaluerer et utrykk ?**
   a) ja b) nei

4. **x og y er to tall. Hvilket av disse utrykkene sjekker om x og y er like ?**
   a) x = y b) x == y c) x != y d) x += y

5. **Hva blir skrevet ut av følgende kode ?**
   a = ‘to ’+’do’
   print(len(a))
   a) 1 b) TypeError c) to do d) 5

6. **Hva blir skrevet ut av følgende kode ?**
   a = ‘5’ + ‘2’
   b = ‘3’ + a
   print(len(b))
   a) 352 b) 3 c) TypeError d) 1

7. **Hva blir skrevet ut av følgende kode ?**
   a = 5 + 5
   ’2’ + a
   print(b)
   a) 255 b) 12 c) TypeError d) 2a

8. **Hva blir skrevet ut av følgende kode ?**
   navn = ‘Per’ + ‘-‘+ ’Olav ’+’Hansen’ ?
   print(navn)
   a) TypeError: unsupported operand type(s) for -: ’str’ and ‘str’
   b) Per-Olav Hansen c) PerOlav Hansen’
   d) SyntaxError: invalid syntax

9. **Hva er resultatet av :** (4 +2) * 7
   a) 42 b) 40 c) 18 d) 28

10. **Hva er resultatet av :** (4 +2) * 7
    a) 40 b) 28 c) 42 d) 18

11. **Hva vil følgende kode returnere ?**

```
x = -3
y = 2
def multiply(x, y):
     """ Return x times y"""
    x *= 4
    return x*y
multiply(x,y)
```

    a) IndentationError b) -6 c) -12 d) 24

12. **Hva vil følgende kode returnere ?**

```
    x = 6
y =  -2
def  add( x, y ):
    ‘’’  Return  x  pluss y ‘’’
       x = 4
    return x + y
         add( x, y )
```

a) 2 b) 4 c) IndentationError d) 6

13. **Hva blir resultatet av denne kode-linjen :**
tekst = ‘5’ + ‘Epler ’ + ‘og’ + 8 + ‘appelsiner’ ?

a) TypeError: Can’t convert ’int’ object to str implicitly
b) tekst == ‘5 epler og 8 apelsiner’
c) tekst =‘5eplerog8appelsiner’
d) SyntaxError: invalid character in identifier

14. **Hva blir resultatet av disse kode-linjene :**
a = 5 + 5
b = 2
c = a + b
a) SyntaxError: can’t assign to literal b) c == ‘a + b’
c) SyntaxError: EOL while scanning string literal d) c== 12

15. **Hva blir verdien til x : x = 36 % 7 :**
a) 2.52 b) 1 c) 2 d) 3

16. **Hva er verdien til x :**
import math
x = math.floor(-0.556) ?

a) -1 b) -0.556 c) 0.556 d) 0

5.2 Results

The pre-test had 25 participants, where 10 indicated they were going to use the app, 11 who would not use it and 4 who didn’t indicate if they were going to use it or not. The results from those who did not indicate if they will use the app were put in the control group, which gives the control group 15 participants. Only one of the participants was familiar with other programming languages, some Java and C#. The same participant did not follow the lectures in INF100, but had started to learn Python to use it in his master degree studies.

The two groups scored about the same in the pre-test. Only 12 participants took the after-test, where eight were from the control group and four were among those who had used the app. When comparing the results from the user-group that used the app with the control group both showed the same progress in the after-test, see Figure 5.1.

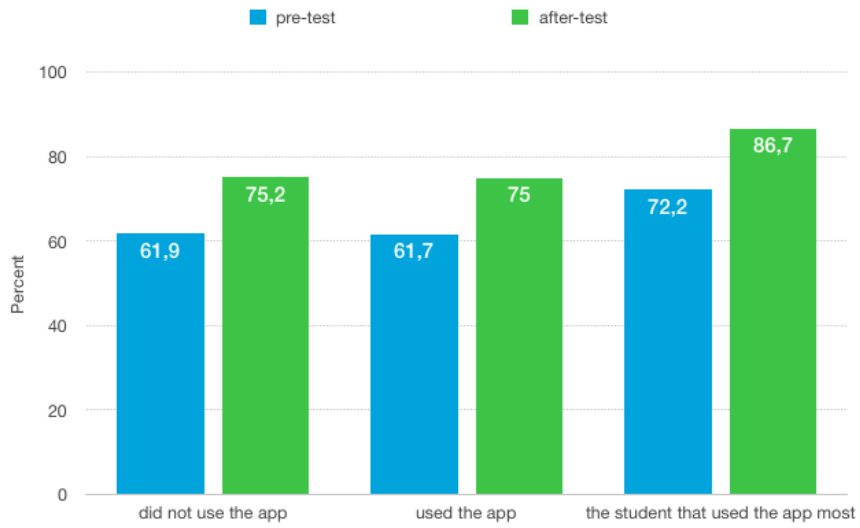


Fig. 5.1 Results from the pre-test and the after-test

The after-test showed that among the group that should use the app, one had used the app 5-10 times, two had used it 3-5 times and one 1-3 times.

When looking at each of the participants in the group using the app, it was the person that had used the app the most, 5-10 times, that scored best out of all participants in the after-test. This student probably had a stronger desire to learn Python than the rest of the group that had the opportunity use the app, and therefore did better in the after test. In the two code interpretation questions in the after-test, the same person was the only person that had both right, and this student had 2 of 4 right in the pre-test.

Question 8 in the after-test turned out to not have any correct answer and was taken away from the results of the after-test. Both groups had answered that question, but the person that had used the app the most had indicated in the answer why she had interpreted the right answer to be a syntax error. This may indicate that by using the app the user learnt that code, and could understand and interpret the output of a code segment. So the app's question of interpreting a small code part may give the user experience in reading and understanding code.

5.3 Conclusion of the user-survey

The results from the user-survey with students taking INF100 of the effectiveness of using the learning app AppiDucks for learning Python can only be used to make observations on how the two groups did. The results need also to be seen in context of how INF100 is taught since AppiDucks is not used as a stand-alone learning app, but as a supplement to INF100. The user-survey does not provide a basis to get a statistically significant result, because there were only 25 participants in the pre-test and 12 in the after test.

In the spring semester of 2019 when the user-survey took place, INF100 had 4 hours with lectures and held 4 hours with groups each week. In the group sessions the students were offered help with their weekly assignments and help to complete 3 mandatory assignments they have to get accepted to take the exam. The course has also given the students the opportunity to submit their weekly assignment to get feedback on their work.

Collaborate learning (CL) is a situation which Wikipedia describes as when two or more people attempt to learn together, and in [24] CL is given an expanded meaning also take place between a student and a teacher in solving problems and complete tasks in the learning process. The fact that INF100 offers the students groups where they can go and get help from a teaching assistant to complete their task gives INF100 an element of collaborative learning in the expanded meaning.

The result from the after-test showed the same progress between both groups from the user-survey. This result is in accordance with other results when it comes to effectiveness of mobile devices for collaborative language learning. Results from other surveys on the effectiveness of mobile devices for collaborative language learning shows a statistically insignificant and small effect [24]. Yet some studies show that mobile devices can support language learners' collaborative learning [24]. A meta-analysis of the effect of using mobile devices on student achievements in language learning [18] showed that the effect of mobile learning was minimal when students took standardized tests. In the meta-analyses they took 20 other studies of the effectiveness of learning apps and compared them to get an average result. The authors of [18] are suspicious there is publication bias in some of the studies, where some results may have been reported as more positive than they really are. They demonstrate this by looking at the difference in results from peer-reviewed papers and from dissertations.

Even though this concerns learning foreign languages and not programming languages in particular, it is still possible to draw parallels to our setting. In addition to the element of CL in INF100, learning a programming language can also be compared to learning a natural language. The documents in Appendix A says that AppiDucks is thought to be implemented using methods inspired from learning foreign languages [17].

The prototype of the app was too unfinished to support any collaborative learning when the user-survey took place and therefore it is not possible to make any conclusion from the user-survey about how the app supports the collaborative learning elements in INF100.

One weakness with the user-survey is that it started in week four and not in the beginning of the semester. This may have been the reason for why the group that should use the app didn't use it that much. When they got the app in week four the tasks may not have been challenging enough compared to assignments they have at group sessions. In [30], a study of 53 825 users of a learning app named 'English Practices' showed that the app was used in average 10 times before the app was uninstalled and that the app was used ca 5 minutes each time it was used. From that it is likely to think that if the users don't find the content in an app relevant enough they will not spend much time on it.

In future work, the content of the app should be expanded to cover the entire syllabus from INF100 and get elements of CL in it. One element could be tasks where students need to work together to complete the tasks. The app also needs to be implemented with a user part that takes care of the user information, like which question types the user has managed to answer correctly. Each question type can be implemented with rules for when the user has managed that type of question. After that she will not get more questions of that type. So when the app starts a new lesson this information is used to create a more personalised lesson with questions that feels relevant for the user. This was something the students in a survey from 2012 [32] mentioned as an advantage of using the learning app. When learning a foreign language the app has the potential of a more personalised learning experience.

When the app has been implemented with all those new features, a new user-survey can be done to see if the new survey will give a different result. This can better show the effectiveness for learning Python between the two groups when the app is used through the semester and has an element of CL in it. In the new user-survey the two user groups should take the pre-test in the first week of the semester. To get more information of how the users are using the app, like how many times they have used it, at what time of the day etc, the app should be made available in Google Play. The authors of [30] concluded that this will provide valuable data regarding the users' behavior by using Google's Firebase Analytics service.

Instead of an after-test, one could compare how the two student groups do at the exam. This will give a more reliable result of whether Appiducks as a supplement to INF100 will help the students to get better results on the exam.

Chapter 6

Conclusion

In this chapter the results of the master thesis are discussed.

The aim of this master thesis has been to look at different approaches in app development and find a suitable framework for developing a learning app for learning Python. After we decided to use a cross-platform solution, the cross-compiled framework Flutter was chosen.

With Flutter we developed a prototype of AppiDucks. Then we evaluated AppiDucks as a supplement in the traditional education of INF100 at UiB, which was the idea behind how AppiDucks was thought to be used. The evaluation was done by conducting a user survey among students following INF100 in the spring semester of 2019.

Some thoughts about the Flutter framework for developing AppiDucks

We decided that a cross-platform solution was the best approach for developing AppiDucks since we wanted to get an app that could run on more than one platform. With the time we had for the development of the app, making a native app for each platform was not an option. The content of the app did not have requirements where a native approach was necessary to give the app better performance.

The Flutter framework was relatively simple to use. The programming language used in Flutter for writing apps is Dart, which has similarities to Java. With a background in Java it was easy to start developing AppiDucks. How simple it feels to start using Flutter is of course something that depends of the developer's background, but with some programming experience it will be relatively easy to start learning.

In the development process we saw a couple of differences of how the the app looked on the iPhone simulator, the Android emulators and phones. The few differences that appeared were often caused by the devices' various screen sizes. This is not something that is caused by Flutter being a cross-platform solution, but is a common problem in app development.

Also, when developing a native app for a particular OS it is hard know how the app will look for the various versions of the devices.

There was one difference, the number keyboard, that for us appeared to be related to the different operating systems. The number keyboard did not pop up in the iPhone simulator, but on the Android emulator it was fine. We may have seen more differences during the development process and with the final result, if the app had used more animation or built-in functionalities in the devices. Since AppiDucks was rather simple we did not have to customize the code for the different OS'es.

When we started to develop AppiDucks with the Flutter framework, Flutter had recently been released as Preview 1 in July 2018 [35]. For that reason its documentation was incomplete and it was hard to find answers for questions that occurred in the development process. Since Flutter was new, the best practice for how to do things had not yet been established, which sometimes made us a bit unsure if we did it the right way under the development.

We were motivated for writing unit and widget tests when we started the development of AppiDucks, but also here the fact that Flutter was released as a preview caused the documentation to be immature. We therefore decided to go for manual testing under the development. It would have been too time-consuming to write unit and widget tests when AppiDucks was relatively easy to test manually.

Overall Flutter is a framework which is easy to use and learn. It is supported by Google. The prototype of AppiDucks is an app with one code base that works on both iPhone and Androids phones. Since we started to develop AppiDucks, more stable versions of Flutter have been released and in addition more device features have been developed as plugins by the Google team. Both the overall documentations about Flutter and how to create widget tests and unit tests have become better the last couple of months.

Evaluating AppiDucks as a learning platform

Learning apps have become a popular tool for learning. It is still a new tool, since apps and smartphones have only been around for about 10 years. For that reason it is necessary with more research in the area, especially of the effect of using learning apps for learning a new subject.

The second part of the master thesis has been to evaluate the effectiveness of using the prototype of Appiducks as a supplement in learning Python, for students following INF100 at UiB. The evaluation of the effectiveness was done by a user survey among students in the spring of 2019. The students were put in two groups, one that could use the app for 4 weeks and a control group.

Both the app and the user survey had some weaknesses in them. The app weakness was that it only covered the three first weeks of the syllabus of INF100 and was not implemented with an analytic and adaptive learning strategy. The user survey started in week 4 and ended in week 8. The survey also had relatively few participants and the app had not been used very much among those who could use it.

Also, the smartphones on their own have weaknesses as a learning tool. The screen size sets limitations for how the app's tasks and its content can be presented on the screen, which may influence the use of learning apps. A survey shows that the average use of the learning app 'English Practices' was 5 minutes [30]. With the smartphone's limitations and the lack of surveys on the effect of using a learning app as a supplement in higher education it may not be a sensible use of resources to develop learning apps for use in higher education at the present time. Since the university has so many other resources for learning, an app may not make a measurable difference for learning a subject.

Future work

Since the use of a learning app is a new way of learning it is important to get more knowledge of the effect of using it and what a good learning app is.

Even though our user survey did not show any effect of using AppiDucks in learning Python for the students taking INF100, more research is needed in the area of using learning apps in traditional teaching. A future work is to implement AppiDucks with the whole syllabus of INF100, with the user survey taking place through the whole semester. This will give more knowledge and more data about the effect, than the user survey of this master thesis which only took place over a short period of time.

In the future work it will also be necessary to look at the technology direction of the app. Should it be more template based, where it is easy to change the content in the app, like the learning platform Kahoot [2]? This solution will make it possible to expand the learning app to other courses. This will again make it easy to do user surveys on other subjects to get more knowledge about the effect of using a learning app in addition to the regular teaching.

Another technology direction for the app is more towards only learning Python. A Python interpreter could be included, which will open up for a new type of exercises. For instance, the students can be given a piece of code that has a bug or two, where the task is to fix the bugs so that the code produces the correct result.

In either of the two possible technology directions of AppiDucks, gamification elements as described in the document "App for å lære" by May-Lill Bagge (see Appendix A) should be implemented in order to get the app more interesting for the students using it.

In a future user survey, the students should start using the app at the beginning of the semester, and the two group's exam results should be compared instead of running an after-test. Then the user survey will show what the effect of using AppiDucks will have on helping the students being prepared for the exam. This, in fact, is the main goal for students who start using a learning app in addition to the traditional education given by the learning institute.

References

[1] Ionic Framework. https://ionicframework.com/docs/, accessed 2019-08-08.

[2] Make learning awesome! [online] https://kahoot.com/, accessed 2019-08-02.

[3] Artsapp enklere artsidentifisering. [online] https://artsapp.uib.no/, accessed 2019-08-09.

[4] Flutter. [online] https://flutter.dev/, accessed 2019-08-09.

[5] [online] https://github.com/rafaelsen/appi_ducks/, saccessed 2019-03-03.

[6] Python tests / quizes. [online] https://pythonspot.com/python-tests-quizes/, accessed 2019-03-03.

[7] The python guru. [online] https://thepythonguru.com/python-guru-quiz/, accessed 2019-03-03.

[8] 1st international conference on learning analytics and knowledge 2011, 2011. [online] https://tekri.athabascau.ca/analytics/, accessed on 2019-08-09.

[9] The nmc horizon report: 2016 higher education edition, 2016. [online] https://www.nmc.org, accessed 2019-08-09.

[10] Platform Architecture, 2018. [online] https://developer.android.com/guide/platform/, accessed 2019-08-08.

[11] Top 10 apps built with Ionic framework, 2018. https://csform.com/top-10-apps-built-with-ionic-framework/, accessed 2019-08-08.

[12] Apps announced at apple's chicago education event, 2018. [online] https://www.cnet.com/pictures/all-the-2018-education-apps-apple-announced/, accessed on 2019-08-09.

[13] Appbrain the go-to place to make your android app successful, 2019. [online] https://flutter.dev/, accessed 2019-08-09.

[14] Smarttelefon, 2019. [online] https://no.wikipedia.org/wiki/Smarttelefon, accessed 2019-08-09.

[15] Arshad Ahmad, Kan Li, Chong Feng, Syed Mohammad Asim, Abdallah Yousif, and Shi Ge. An empirical study of investigating mobile applications development challenges. *IEEE Access*, 6:17711 – 17728, 2018.

[16] Andreas Biørn-Hansen, Tor-Morten Grønli, and Gheorghita Ghinea. A survey and taxonomy of core concepts and research challenges in cross-platform mobile development. *ACM Computing Surveys*, 51(5):1–34, 2018.

[17] Pat Byrne and Gerry Lyons. The effect of student attributes on success in programming. In *Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '01, pages 49–52. ACM, 2001.

[18] Kyunghwa Cho, Sungwoong Lee, Min-Ho Joo, and Betsy Jane Becker. The effects of using mobile devices on student achievement in language learning: A meta-analysis. *Education Sciences*, 8(3), 2018.

[19] Matteo Ciman and Ombretta Gaggi. An empirical analysis of energy consumption of cross-platform frameworks for mobile development. *Pervasive and Mobile Computing*, 39:214 – 230, 2017.

[20] Robert Godwin-Jones. Emerging technologies: Mobile apps for language learning. *Language Learning and Technology*, 15:2–11, 2011.

[21] Daniel Grenier. The relationship between mobile learning and academic achievement in a community college system online environment. 2018.

[22] Philip Kerr. Adaptive learning. *ELT Journal*, 70(1):88–93, 2015.

[23] Anmol Khandeparkar, Rashmi Gupta, and B.sindhya. An introduction to hybrid platform mobile application development. *International Journal of Computer Applications*, 118(15):31–33, 2015.

[24] Agnes Kukulska-Hulme and Olga Viberg. Mobile collaborative language learning: State of the art. *British Journal of Educational Technology*, 49(2):207–218, 2018.

[25] Kevin B. Larkin. Mathematics education: Is there an app for that?. 2013.

[26] Max Lynch. Basic unit testing in ionic, 2017. [online] https://blog.ionicframework.com/basic-unit-testing-in-ionic/, accessed 2019-08-09.

[27] Johannes C. Cronje Mohamed Osman M. El-Hussein. Defining mobile learning in the higher education landscape. In *Proceedings of the 7th International Workshop on Automation of Software Test*, pages 12–21. International Forum of Educational Technology Society, 2010.

[28] Nachiappan Nagappan, E. Michael Maximilien, Thirumalesh Bhat, and Laurie Williams. Realizing quality improvement through test driven development: results and experiences of four industrial teams. In Pankaj Jalote, editor, *Springer Science + Business Media, LLC 2007*. Spinger, 2008.

[29] Gloria Omale. Gartner says demand for top chinese brands drove worldwide smartphone sales in third quarter 2018, 2019. [online] https://www.gartner.com/en/newsroom/press-releases/2018-12-03-gartner-says-demand-for-top-chinese-brands-drove-worl, accessed 2019-08-09.

[30] Xuan Lam Pham, Thi Huyen Nguyen, and Gwo Dong Chen. Research through the app store: Understanding participant behavior on a mobile english learning app. *Journal of Educational Computing Research*, 56(7):1076–1098, 2018.

[31] Sophia Shing and Benjamin Yuan. Apps developed by academics. *Journal of Education and Practice*, 7(33), 2016.

[32] Caroline Steel. Fitting learning into life: Language students' perspectives on benefits of using mobile apps. In M. Brown, M. Hartnett, and T. Stewart, editors, *ASCILITE 2012*, pages 875 – 880. Australasian Society for Computers in Learning in Tertiary Education, 2012.

[33] Techotopia. The iphone os architecture and frameworks, 2016. [online] https://www.techotopia.com/index.php/The_iPhone_OS_Architecture_and_Frameworks, accessed 2019-08-09.

[34] Wikipedia. Test-driven Development, 2018. [online] https://en.wikipedia.org/wiki/Test-driven_development, accessed 2019-08-08.

[35] Wikipedia. Flutter (software), 2018. [online] https://en.wikipedia.org/wiki/Flutter_(software), accessed 2019-08-09.

[36] Wikipedia. Ionic (mobile app framework), 2018. https://en.wikipedia.org/wiki/Ionic_(mobile_app_framework), accessed 2019-08-08.

[37] Wikipedia. Jasmine (javascript testing framework), 2018. [online] https://en.wikipedia.org/wiki/Jasmine_(JavaScript_testing_framework), accessed 2019-08-08.

[38] Wikipedia. iOS SDK, 2018. [online] https://en.wikipedia.org/wiki/IOS_SDK, accessed 2019-08-08.

[39] Jeremy Wilken. What you need to know to start building mobile apps with the ionic framework, 2014. [online] https://gnomeontherun.com/2014/10/18/what-you-need-to-know-to-start-building-mobile-apps-with-the-ionic-framework/, accessed 2019-08-08.

[40] Wkipedia. Android software development. [online] https://en.wikipedia.org/wiki/Android_software_development, accessed 2019-08-08.

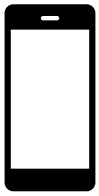
[41] Friedel Ziegelmayer. Things should be simple. we believe in testing and so we want to make it as simple as possible. [online] https://karma-runner.github.io/2.0/index.html, accessed 2019-08-09.

Appendix A

content of the app

APP FOR Å LÆRE PYTHON

MAY-LILL BAGGE OG
SOLVEIG RADDUM



HVA APPEN SKAL GJØRE

Appen benytter metoder fra å lære fremmedspråk og er ment som en ekstra hjelp på et kurs i Python. Det er ikke ment å være en frittstående opplæring i programmering.

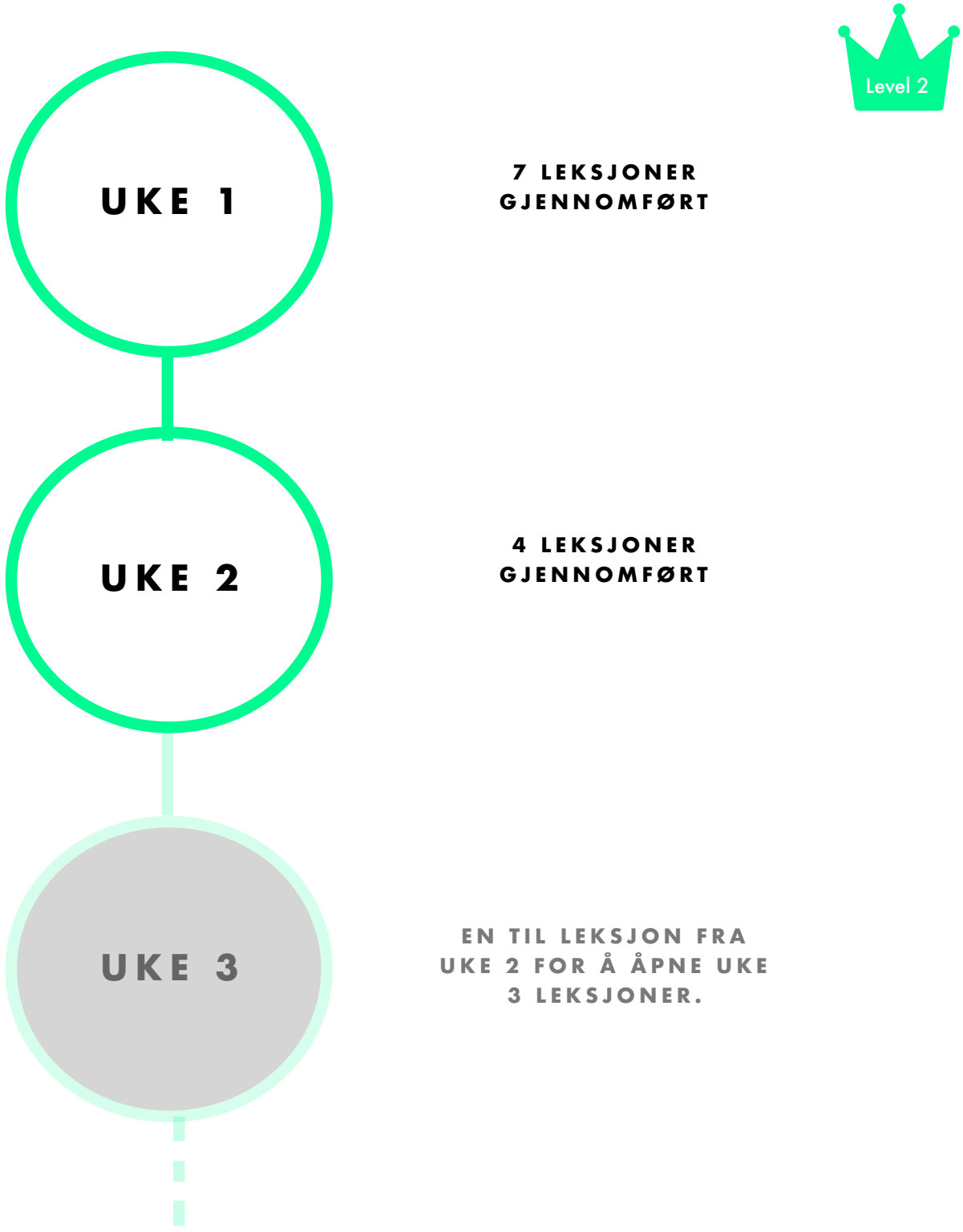
Appen er videre forklart utover i dette dokumentet.

Appen er delt inn i leksjoner.

Først forklares hvordan leksjonene er organisert i forhold til hverandre og hvordan brukere kommer til de forskjellige leksjonene og lignende, deretter kommer forklaringen på hvordan selve lesksjonene er organisert. Så kommer en forklaring på spørsmålstypene. Til sist kommer de forskjellige leksjonene.

ÅPNINGS SKJERM

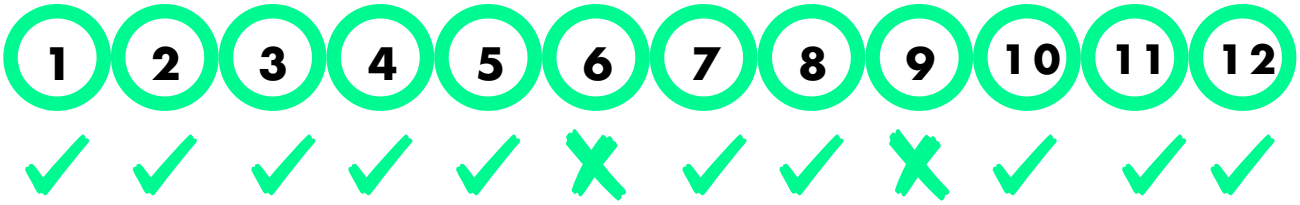
Appen er ment å bruke sammen med et tolv ukers kurs. Organiseringen av leksjoner er delt inn i disse tolv ukene + et sett med leksjoner for eksamensforberedelse etter dette.



LEKSJONER

Hver leksjon består av 12 spørsmål. Selve spørsmålene for hver uke er beskrevet senere. Hvordan leksjonene inkorpororerer skryt og noen flere andre egenskaper ved leksjonene er bekrevet her.

EKSEMPEL



En student gjennomfører en leksjon med 12 spørsmål, og etter de første tre rette får han tilbakemelding fra appen. Det samme etter de første fem som også er rette.

Det vil også være skryt etter ti og tolv rette.

Liste med faktiske komplimenter som appen kan gi:

- Tre rette på rad! Hardt arbeid lønner seg.
- Fem rette på rad! Nå begynner du å bli en mester i dette.
- Ti rette på rad! Nå begynner du å bli klar til eksamen i materialet fra denne uken.
- Tolv rette på rad! Hm, kanskje du skal øve på en av de andre ukene? Dette kan du allerede. Dette bør åpne for neste uke om den ikke allerede er åpnet.

Etter de tolv spørsmålene er gjennomført, vil studenten få spørsmål 6 og 9 om igjen før leksjonen er over. Alle spørsmålene må besvares rett før leksjonen er over, uansett hvor mange forsøk man trenger på det.

HVERT SPØRSMÅL

Når en student blir presentert med et spørsmål, vil han eller hun ha den tiden hun måtte ønske på å svare. Et svar er enten rett eller galt.

Etter et svar er gitt vil det komme en tilbakemelding og en knapp til å gå til neste.

Neste fører til neste spørsmål, untatt ved siste spørsmål, hvor det fører til en oppsummering. Den vil bestå av hvor mye man har jobbet og fokusere på innsats heller enn antall rette.

OPPSUMMERING

Etter hver leksjon kommer en oppsummering som gir tilbakemelding om hvor mye man har jobbet i det siste. Dette gir studenten økende level. Kanskje disse kan ha forskjellige symboler, eller en student kan ha et tilfeldig valgt symbol.

Level vil aldri gå nedover, utelukkende oppover. Et level for tre dager på rad, et ekstra level for 7 dager på rad, et ekstra for 14 dager på rad og så oppover.

Neste her vil føre tilbake til valg av uke.

DATA

Hver student må lage seg en bruker med brukernavn og passord.

Når brukeren har gjort dette kan appen spørre om studenten tar et kurs i programmering med Python på UiB. I så tilfelle kan den spørre hvilket og om det er ok å lagre noen data rundt studentens bruk av appen til forskning. For å kunne lagre data om brukere må det søkes til datatilsynet, og det bør lages en side i appen som forklarer hvilke data som blir tatt vare på.

Data det kan være greit å ta vare på om brukerne er:

Når de bruker appen, hvor mye de svarer rett ved hver sesjon, hvor mange leksjoner i hver sesjon, og en selvrapportert karakter ved slutten av semesteret. Ellers ingen statistiske data som kjønn og alder.

Det bør også lages en copyright side, antageligvis.

SPØRSMÅLSTYPER

Det vil være flere typer spørsmål i hver av ukene, hvor en database av spørsmål vil by på hver enkelt leksjon.

Hver uke vil ha en gitt fraksjon av hvert spørsmål/spørsmålstype som skal sette sammen leksjonene.

EKSEMPEL

Uke A (Dette er ikke en faktisk uke i appen.)

Spørsmål type 1 - 1 spm

Spørsmål type 2 om emne X - 2 spm

Spørsmål type 2 om emne Y - 3 spm

Spørsmål type 3 om emne X - 4 spm

Spørsmål type 3 om emne Y - 2 spm

Hver av disse ukefraksjonene blir gitt senere i dette dokumentet.

SPØRSMÅL TYPE A

Disse spørsmålene er alle skrevet fullstendig i dette dokumentet og spør etter betydningen av nøkkelord i Python. Spørsmålene her blir gitt som fire ord med fire mulige betydninger, hvorav et ord er rett og en betydning er rett. Spørsmålene kan slik spørres begge veier. De to rette står i bold.

EKSEMPEL

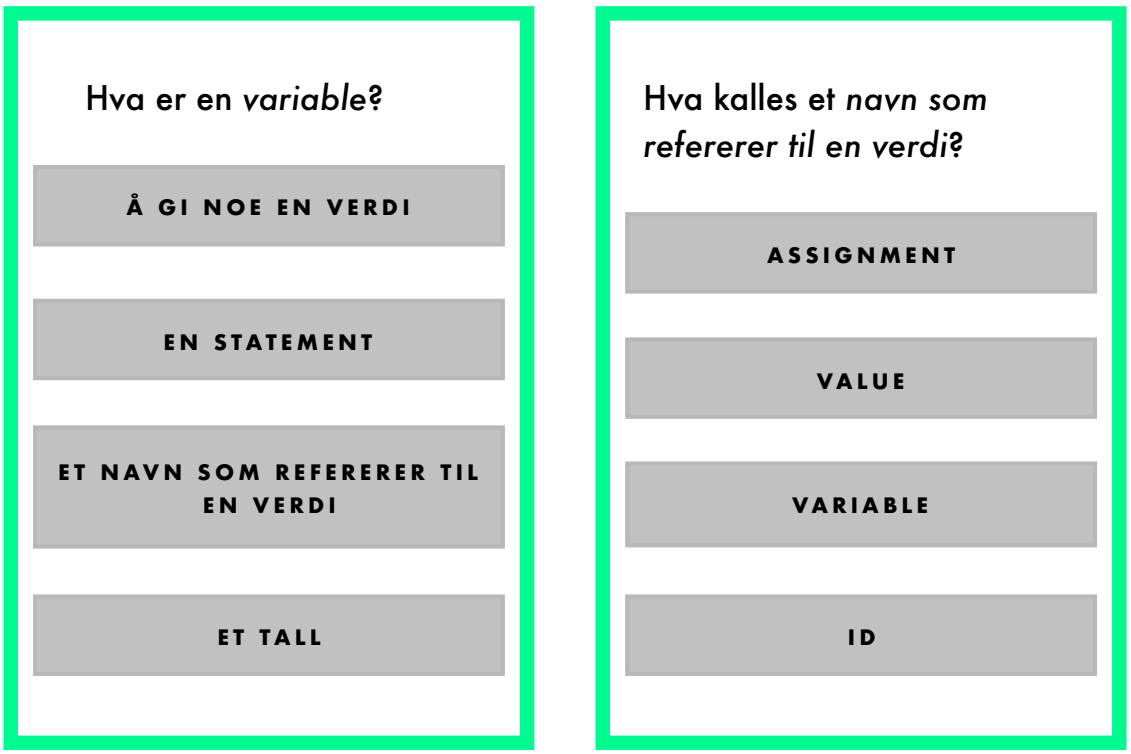
Ord:

Variable, assignment, value, id.

Forklaringer:

Et tall, å gi noe en verdi, **et navn som refererer til en verdi**, en statement.

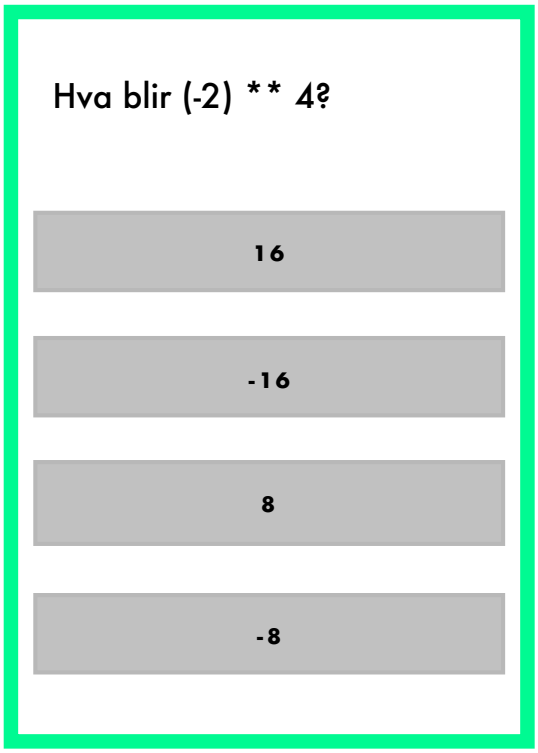
Dette kan så generere to spørsmål:



SPØRSMÅL TYPE B

Disse spørsmålene genererers av appen basert på et sett med regler. Reglene blir gitt i sin helhet i dette dokumentet.

EKSEMPEL



Regelen som har produsert dette spørsmålet er ganske basic i alle programmeringsspråk og trenger kanskje ikke så mye forklaring.

Feks kan det forklares slik:

Precedence	Operator	Operation
Highest	**	Exponentiation
	-	Negation
	*, /, //, %	Multiplication, division, integer division, and remainder
Lowest	+, -	Addition and subtraction

Plukk tilfeldige tall mellom -15 og 15, men ikke null, til å teste studentene i disse reglene.

Etter den sjette gangen en student får til spørsmål om et emne på første forsøk kan de godt bli bedt om å taste inn svaret uten alternativer. Dette kan gå over flere uker. Dvs en type spørsmål som gjentaes i senere uker, kan bruke oppsummert score på spørsmålstypen uavhengig av uke. Denne counteren for tasting versus alternativer er egen ved repetisjon, og egen i hver uke den er med på repetisjon.

EKSEMPEL

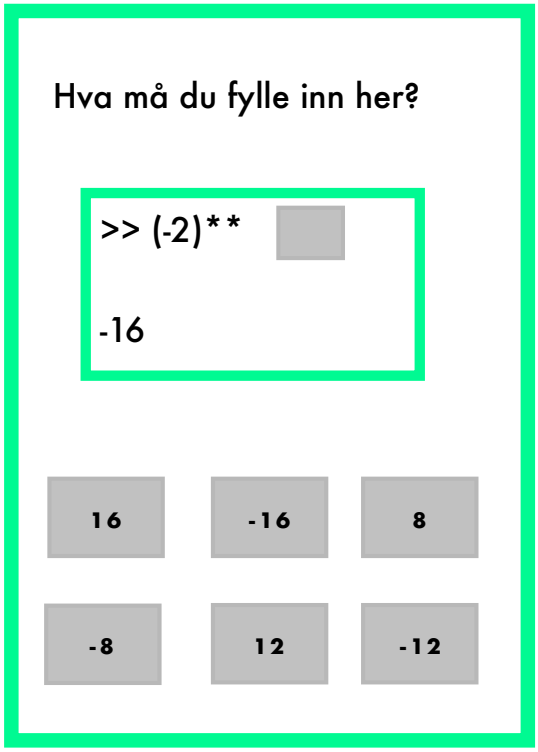
Dette bør ikke settes i gang ved gjenntagelse av spørsmål som brukeren ikke har fått til, de bør være av helt samme form når de gjentaes. Dette kan settes i gang mindt i en leksjon. Dvs. en leksjon kan ha et spørsmål om et tema med alternativer og et med fri tekst.

Svaralternativene kan genereres med noen tilfeldige alternativer og noen tilfeller hvor en regel blir brukt feil. I de tilfellene hvor det ikke er mulig å generere feile alterntiver bruker vi fri tekst til å besvare spørsmålet hele veien.

SPØRSMÅL TYPE C

Denne typen spørsmål er også generert av et sett med regler som kommer videre i dokumentet. Det vil gjerne være de samme reglene som i type B spørsmålene.

EKSEMPEL



Ved å trykke på en av alternativene vil det flytte seg opp til det tomme feltet. Brukeren vil da ha to valg. Enten å trykke på det grå feltet og få alternativet til å gå tilbake til listen over alternativ, eller å trykke på Sjekk svaret og på den måten gå videre i leksjonen.

Også her vil bare de seks første oppgavene innen et tema gi alternativer. Etter det vil man bli bedt om å taste inn svaret. (Vil være utformet som på spørsmålstype B.)

BADGES

Se mozilla bagdes (google it). De ser ut til å være open source og slikt.

Brukerne vil få et bage etter de har logget inn og gjort en leksjon for første gang. Dette kan kalles **Newbie**.

Brukerne får badges basert på tid på dagen hvor appen brukes jevnlig.

Morgen	6-12	Earlybird
Ettermiddag	12-18	Daydweller
Kveld	18-24	Eveningstar
Natt	24-6	Nightowl

Fem dager på rad med minst en leksjon innen den samme av disse tidene gir deg badgen i bronse. Det samme igjen gir deg badgen i sølv, og det samme igjen gir deg badgen i gull.

Brukerne får også badges basert på hvor mye de har øvd.

Antall leksjoner	
10	Explorer - bronse
25	Explorer - sølv
50	Explorer - gull
75	Adventurer - bronse
100	Adventurer - sølv
125	Adventurer - gull
150	Superstar - bronse
175	Supertar - sølv
200	Superstar - gull

De forskjellige kategoriene som spørsmålene tilhører gir også badges. (Flere kommer.)

Type	**Bagde**
Matte	Mathemathician
Typer	Typer
Variabler	Variabler
Feil	Error finder
Funksjoner	Functioner

Type	Bagde
Test	Tester
Strenger	Strenger
Print	Printer
Boolean	Booler
If	Iffer
Modul	Modular
Kommentarer	Commentator
Metoder	Methodician
Liste	Lister
For	Looper
While	Whiler
Objekter	Objector

Type bagdene kan oppnåes i bronse, sølv og gull. 10 spørsmål rett på første forsøk i en kategori gir bronse, 25 gir sølv og 50 gir gull.

Brukerne kan også få **twinsies** badges. Dette er den eneste typen badges som ikke vises som grået ut i badges skjermen. De vil bare dukke opp når det faktisk oppnåes. Alle de andre typene badges vil vise som grået ut når ikke det laveste level er oppnådd. Ved å klikke på et av badgene vil man få opp hva som kreves for å oppnå det. For twinsies vil det å klikke på badgen gi deg et (eller flere) brukernavn som du har oppnådd twinies med.

Twinsies oppnåes når to brukere har gjennomført tre leksjoner likt, og er på samme hovednivå i antall leksjoner (dvs er enten explorer, adventurer eller supertar, avhengig av metallet) når de gjør leksjonen likt.

Å gjennomføre en leksjon likt er at en leksjon fra samme uke har de samme spørsmålene mestret, det kan være at ingen er mestret. Og at de får de samme spørsmålstypene rett på første forsøk, og eventuelle feil som oppstår i gjenntagelsen av feile spørsmål i leksjonen også er like. (Se ellers delen om leksjoner.)

Eksempel:

	1. forsøk	2. forsøk	3. forsøk
1	Rett		
2	Rett		
3	Ikke rett	Rett	
4	Rett		

	1. forsøk	2. forsøk	3. forsøk
5	Rett		
6	Ikke rett	Ikke rett	Rett
7	Rett		
8	Rett		
9	Rett		
10	Rett		
11	Ikke rett	Rett	
12	Rett		

PENSUM

Pensum er hentet fra boken “Practical Programming - An Introduction to Computer Science Using Python” Third Edition av Paul Gries, Jennier Campbell og Jason Montojo.

Uke 1 - Kapittel 1 og 2

Uke 2 - Kapittel 3

Uke 3 - Kapittel 15

Uke 4 - Kapittel 4

Uke 5 - Kapittel 5

Uke 6 - Kapittel 6

Uke 7 - Kapittel 7

Uke 8 - Kapittel 8

Uke 9 - Kapittel 9.1 til 9.5

Uke 10 - Resten av kapittel 9

Uke 11 - kapittel 10

Uke 12 - Kapittel 14

Ekstrauke - Repetisjon av alt - Under beskrivelsen av denne uken vil også være noe om hva som alt er mestret og hvordan man forholder seg til repetisjon av det. Videre beskrives hver uke separat med de forskjellige spørsmålene.

Spørsmålsnumerering og notasjon

Hver uke har et nummer (ekstrauken vil ikke ha egne spørsmålstyper, bare repetisjon av tidligere, og trenger derfor ikke et unikt nummer), og hver spørsmålstype fra en uke vil telle fra 1 og oppover. Slik vil alle spørsmålstypemne få en unik iD - feks. spørsmålstype 1 fra uke 1 heter 1.1 og spørsmålstype 5 fra uke 6 heter 6.5. På slutten av beskrivelsen til hvert spørsmål vil det få en kategori. Dette vil benyttes senere til badges.

Når alt innen en spørsmålstype er mestret, vil spørsmålene som til å begynne med er av denne typen, gå til de andre typene i den uken som enda ikke er mestret tilfeldig fordelt.

Når alt i en uke er mestret, vil det stå ved valget av denne uken at alle spørsmålene her er mestret, mens det fremdeles er mulig å velge uken og alle spørsmålstypene er da tilgjengelig igjen. Man vil ikke da ha muligheten til å mestre noe i denne uken. Spørsmålene vil da være tilgjengelig for repetisjon i andre uker.

Generelt om mestring

For spørsmål av type A - hver av de to variantene av et spørsmål må besvares rett 2 ganger for at denne varianten skal mestres. I tabellene under er det gitt mestring for andre typer spørsmål. Type 1 i alle ukene har bare type A spørsmål. Mestring av hver type spørsmål er gitt ved sin spørsmålstype. Dette gjender da for spørsmål av type B og C.

Repetisjonsspørsmål i senere uker teller mot mestring i sin origianle uke om det ikke alt er oppnådd. Dette gjelder generelt.

UKE 1 (KAPITTEL 1 OG 2)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 3, 7 og 10.

Spørsmålstype 2 - Evaluering av basal matematikk. Kategori: matte - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 7 og 10.

Spørsmålstype 3 - Typer. Kategori: typer - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 7 og 10.

Spørsmålstype 4 - Operator presedens. Kategori: matte- 3 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 2, 7 og 10.

Spørsmålstype 5 - Numerisk presisjon. Kategori: matte - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 7 og 10.

Spørsmålstype 6 - Variabler og assignment. Kategori: variabler - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 3, 7 og 10.

Spørsmålstype 7 - Augmented assignment. Kategori: variabler - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 7 og 10.

Spørsmålstype 8 - Syntax errors. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 7 og 10.

Uke 1 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
1.1 - Spørsmål	Diverse	1		3, 7, 10
1.2 - Matte operasjoner	Matte	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 7, 10
1.3 - Typer	Typer	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 7, 10
1.4 - Operator presendens	Matte	3	Hver regel - 10 ganger rett og 3 rett på rad	2, 7, 10
1.5 - Numerisk presisjon	Matte	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 7, 10
1.6 - Variabler og assignment	Variabler	2	Hver regel - 10 ganger rett og 5 rett på rad	3, 7, 10
1.7 - Augmented assignment	Variabler	2	Hver regel - 10 ganger rett og 3 rett på rad	4, 7, 10
1.8 - Syntax errors	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 7, 10

UKE 2 (KAPITTEL 3)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 2 - Spørsmål om innebygde funksjoner. De diverse funksjonene kan ha egen kategori. (Hovedsakelig matte.) - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 4, 8 og 11.

Spørsmålstype 3 - små funksjoner. Kategorier vil være egne. (Hovedsakelig matte.) - 3 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 3, 8 og 11.

Spørsmålstype 4 - feilmeldinger. Kategori: feil. - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 5 - kommentarer. Kategori: kommentarer. - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 6 - tester. Kategori: test. - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 7 - navn. Kategori: vaiabler. 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 8 - repestisjon. Kategori: varierer - 1 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. Det er bare fra 1.4 i denne uken, så de er alle likt vektet og kan fritt velges randomly av programmet. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 2 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
2.1 - Spørsmål	Diverse	1		5, 8, 11
2.2 - Innebygde funksjoner	Diverse	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 8, 11
2.3 - Funksjoner	Diverse	3	Hver regel - 10 ganger rett og 5 rett på rad	3, 8, 11
2.4 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
2.5 - Kommentarer	Kommentarer	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
2.6 - Tester	Test	2	Hver regel - 10 ganger rett og 3 rett på rad	5, 8, 11

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
2.7 - Navn	Variabler	2	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
2.8 - Repetisjon	Diverse	1	Repetisjon denne uken: 1.4	

UKE 3 (KAPITTEL 15)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 2 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 2 - tester for det som har vært hittil. Kategori: test, eventuelt en del andre kategorier - 7 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 3 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 4 - repestisjon. Kategori: varierer - 2 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 1.1 og 2.3 repeteres denne uken. De to spørsmålene kan velges tilfeldig fra de to spørsmålstypene med 1.1 vekt 1 og 2.3 vekt 2. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 3 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
3.1 - Spørsmål	Diverse	2		6, 9, 12
3.2 - Tester	Test	7	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
3.3 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
3.4 - Repetisjon	Diverse	2	Repetisjon denne uken: 1.1 og 2.3	

UKE 4 (KAPITTEL 4)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 2 - Tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 3 - Strengoperasjoner. Kategori: strenger - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 4 - Printe. Kategori: print - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 5 - Tastaturet. Kategori: strenger - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 6 - feilmeldinger nye i dette kapittelet. Kategori: feil - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 5, 8 og 11.

Spørsmålstype 7 - repestisjon. Kategori: varierer - 2 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 1.2, 1.3, 1.5, 1.7, 1.8, 2.2 repeteres denne uken. De to spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 4 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
4.1 - Spørsmål	Diverse	1		5, 8, 11
4.2 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
4.3 - Strengoperasjoner	Strenger	2	Hver regel - 10 ganger rett og 5 rett på rad	5, 8, 11
4.4 - Printe	Print	2	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
4.5 - Tastatur	Strenger	2	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11
4.6 - Feilmeldinger	Feil	2	Hver regel - 10 ganger rett og 3 rett på rad	5, 8, 11
4.7 - Repetisjon	Diverse	2	Repetisjon denne uken: 1.2, 1.3, 1.5, 1.7, 1.8, 2.2	

UKE 5 (KAPITTEL 5)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 2 - boolean typer. Kategori: boolean - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 3 - boolean operatorerer. Kategori: boolean - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 4 - relasjonelle operatorer. Kategori: matte - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 5 - blandede operatorer. Kategori: boolean - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 6 - sammenligne strenger. Kategorier: strenger - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 7 - if setteninger. Kategori: if - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 8 - elif setninger. Kategori: if - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 9 - else settninger. Kategori: if - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 10 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 11 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 6, 9 og 12.

Spørsmålstype 12 - repestisjon. Kategori: varierer - 1 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 2.1, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 repeteres denne uken. Spørsmålet kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 5 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
5.1 - Spørsmål	Diverse	1		6, 9, 12
5.2 - Boolean typer	Boolean	1	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
5.3 - Boolean operatorer	Boolean	1	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
5.4 - Relasjonelle operatorer	Matte	1	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
5.5 - Blandede operatorer	Boolean	1	Hver regel - 10 ganger rett og 5 rett på rad	6, 9, 12
5.6 - Sammenligne strenger	Strenger	1	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12
5.7 - If settninger	If	1	Hver regel - 10 ganger rett og 5 på rad	6, 9, 12
5.8 - Elif settninger	If	1	Hver regel - 10 ganger rett og 5 på rad	6, 9, 12
5.9 - Else	If	1	Hver regel - 10 ganger rett og 5 på rad	6, 9, 12
5.10 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 på rad	6, 9, 12
5. 11 - Tester	Test	1	Hver regel - 5 ganger rett og 3 på rad	6, 9, 12
5.12 - Repetisjon	Diverse	1	Repetisjon denne uken: 2.1, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6	

UKE 6 (KAPITTEL 6)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 2 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 7 og 10.

Spørsmålstype 2 - bruk av modulene importert i kapittelet. Kategori: modul - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 7 og 10.

Spørsmålstype 3 - bruk av egne moduler. Kategori: modul - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 7 og 10.

Spørsmålstype 4 - feilmeldinger nye i dette kapittelet. Kategori: feil - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 7 og 10.

Spørsmålstype 5 - tester for det nye i kapittelet. Kategori: test - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 7 og 10.

Spørsmålstype 6 - repestisjon. Kategori: varierer - 3 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 3.1, 3.2, 3.3, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.7, 5.9, 5.10, 5.11 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 6 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
6.1 - Spørsmål	Diverse	2		*7*, 10
6.2 - Importerte moduler	Moduler	1	Hver regel - *5* ganger rett og 3 rett på rad	*7*, 10
6.3 - Egne moduler	Moduler	2	Hver regel - *5* ganger rett og 3 rett på rad	*7*, 10
6.4 - Feilmeldinger	Feil	2	Hver regel - *5* ganger rett og 3 rett på rad	*7*, 10
6.5 - Tester	Test	2	Hver regel - *5* ganger rett og 3 rett på rad	*7*, 10
6.6 - Repetisjon	Diverse	3	Repetisjon denne uken: 3.1, 3.2, 3.3, *5.1*, *5.2*, *5.3*, *5.4*, *5.5*, *5.6*, *5.7*, *5.7*, *5.9*, *5.10*, *5.11*	

UKE 7 (KAPITTEL 7)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 2 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 2 - metodekall. Kategori: metoder - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 3 - streng metoder. Kategori: metoder - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 4 - underscores. Kategori: metoder - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 5 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 6 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 9 og 11.

Spørsmålstype 7 - repestisjon. Kategori: varierer - 3 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 6.1, 6.2, 6.3, 6.4, 6.5 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 7 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
7.1 - Spørsmål	Diverse	2		9, 11
7.2 - Metodekall	Metoder	2	Hver regel - 5 ganger rett og 3 rett på rad	9, 11
7.3 - Streng metoder	Metoder	2	Hver regel - 5 ganger rett og 3 rett på rad	9, 11
7.4 - Underscores	Metoder	1	Hver regel - 5 ganger rett og 3 rett på rad	9, 11
7.5 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	9, 11
7.6 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	9, 11
7.7 - Repetisjon	Diverse	3	Repetisjon denne uken: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 6.1, 6.2, 6.3, 6.4, 6.5	

UKE 8 (KAPITTEL 8)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 2 - lister. Kategori: liste - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 3 - endre lister. Kategori: liste - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 4 - aliaser. Kategori: variables - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 5 - liste metoder. Kategori: metoder - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 6 - lister av lister. Kategori: lister - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 7 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 8 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 9 - repestisjon. Kategori: varierer - 3 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 8 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
8.1 - Spørsmål	Diverse	1		10, 12
8.2 - Liste	Liste	1	Hver regel - 5 ganger rett og 3 rett på rad	10, 12
8.3 - Endre liste	Liste	2	Hver regel - 10 ganger rett og 5 rett på rad	10, 12
8.4 - Aliaser	Variabler	1	Hver regel - 5 ganger rett og 3 rett på rad	10, 12
8.5 - Listemetoder	Metoder	1	Hver regel - 10 ganger rett og 5 rett på rad	10, 12
8.6 - Lister av lister	Liste	1	Hver regel - 10 ganger rett og 5 rett på rad	10, 12
8.7 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	10, 12
8.8 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	10, 12
8.9 - Repetisjon	Diverse	3	Repetisjon denne uken: 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6	

UKE 9 (KAPITTEL 9.1 TIL 9.5)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 2 - for loop. Kategori: for - 2 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst femten ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10, 11 og 12.

Spørsmålstype 3 - liste traverserig. Kategori: for - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 4 - range. Kategori: liste - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 5 - nøstede for loops. Kategori: for - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst femten ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10, 11 og 12.

Spørsmålstype 6 - nøstede lister. Kategori: liste - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i ukene 10 og 12.

Spørsmålstype 7 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 8 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 9 - repestisjon. Kategori: varierer - 3 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 3.1, 3.2, 3,3, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 9 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
9.1 - Spørsmål	Diverse	1		11
9.2 - For loop	For	2	Hver regel - 15 ganger rett og 5 rett på rad	10, 11, 12
9.3 - Liste traversering	For	1	Hver regel - 10 ganger rett og 3 rett på rad	10, 12
9.4 - Range	Liste	1	Hver regel - 5 ganger rett og 3 rett på rad	11
9.5 Nøstede for loops	For	1	Hver regel - 15 ganger rett og 5 rett på rad	10, 11, 12
9.6 Nøstede lister	Liste	1	Hver regel - 10 ganger rett og 5 rett på rad	10, 12
9.7 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	11
9.8 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	11
9.9 - Repetisjon	Diverse	3	Repetisjon denne uken: 3.1, 3.2, 3,3, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6	

UKE 10 (RESTEN AV KAPITTEL 9)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 2 - while loop. Kategori: while - 3 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst femten ganger og minst fem av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 3 - break. Kategori: while - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 4 - continue. Kategori: while - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst ti ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 5 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 6 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 11.

Spørsmålstype 7 - repestisjon. Kategori: varierer - 4 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 6.1, 6.2, 6.3, 6.4, 6.5, 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 9.2, 9.3, 9.5, 9.6 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 10 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
10.1 - Spørsmål	Diverse	1		11
10.2 - While loop	While	3	Hver regel - 15 ganger rett og 5 rett på rad	12
10.3 - Break	While	1	Hver regel - 10 ganger rett og 3 rett på rad	12
10.4 - Continue	While	1	Hver regel - 10 ganger rett og 3 rett på rad	12
10.5 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	11
10.6 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	11
10.7 - Repetisjon	Diverse	4	Repetisjon denne uken: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 6.1, 6.2, 6.3, 6.4, 6.5, 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 9.2, 9.3, 9.5, 9.6	

UKE 11 (KAPITTEL 10)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori. - 1 spm

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 2 - with as. Kategori: streng - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 3 - for line in. Kategori: for - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 4 - write. Kategori: streng - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 5 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 6 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Repetisjon: Uavhengig av mestring blir det en kandidat for repetisjon i uke 12.

Spørsmålstype 7 - repestisjon. Kategori: varierer - 6 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.4, 4.5, 4.6, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 9.1, 9.2, 9.4, 9.5, 9.7, 9.8, 10.1, 10.5, 10.6 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 11 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring	Repetisjon
11.1 - Spørsmål	Diverse	1		12
11.2 - With as	Streng	1	Hver regel - 5 ganger rett og 3 rett på rad	12
11.3 - For line in	For	1	Hver regel - 5 ganger rett og 3 rett på rad	12
11.4 - Write	Streng	1	Hver regel - 5 ganger rett og 3 rett på rad	12
11.5 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	12
11.6 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad	12
11.7 - Repetisjon	Diverse	6	Repetisjon denne uken: 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 4.1, 4.2, 4.4, 4.5, 4.6, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 9.1, 9.2, 9.4, 9.5, 9.7, 9.8, 10.1, 10.5, 10.6	

UKE 12 (KAPITTEL 14)

Spørsmålstype 1 - Spørsmål om nøkkelord i kapittelet med gitte svaralternativer. Hver av disse spørsmålene kan ha egen kategori.

Disse spørsmålene vil være av type A.

Mestring: Etter at et av disse spørsmålene er besvart rett på første forsøk to ganger (dvs. en av de to spørsmålene laget fra et sett med fire alternativer med spørsmål og svar) vil de ikke komme opp igjen i denne ukens leksjoner. Den motsatte typen kan fremdeles dukke opp.

Spørsmålstype 2 - objekt orientering. Kategori: objekter

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Spørsmålstype 3 - feilmeldinger nye i dette kapittelet. Kategori: feil - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Spørsmålstype 4 - tester for det nye i kapittelet. Kategori: test - 1 spm

Disse spørsmålene kan være av alle tre typene.

Mestring: Når en regel for evaluering i denne spørsmålstypen er svart korekt på første forsøk minst fem ganger og minst tre av disse er korekt på rad (uten noen feil i denne typen i mellom) vil spørsmål om den regelen ikke komme oppp igjen i denne uken.

Spørsmålstype 5 - repestisjon. Kategori: varierer - 8 spm

Spørsmål fra tidligere uker som er tilgjengelig for repetisjon i denne uken. 3.1, 3.2, 3.3, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 9.2, 9.3, 9.5, 9.6, 10.2, 10.3, 10.4, 11.1, 11.2, 11.3, 11.4, 11.5, 11.6 repeteres denne uken. Spørsmålene kan velges tilfeldig fra alle spørsmålstypene med lik vekt på alle typene. Emner og spørsmål skal repeteres uavhengig av mestring.

Uke 12 Oppsumering

Spmørsmålstype	Kategori	# spørsmål	Mestring
11.1 - Spørsmål	Diverse	1	
11.2 - Objekt orientering	Objekter	1	Hver regel - 5 ganger rett og 3 rett på rad
11.5 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad
11.6 - Tester	Test	1	Hver regel - 5 ganger rett og 3 rett på rad
11.7 - Repetisjon	Diverse	8	Repetisjon denne uken: 3.1, 3.2, 3.3, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 9.2, 9.3, 9.5, 9.6, 10.2, 10.3, 10.4, 11.1, 11.2, 11.3, 11.4, 11.5, 11.6

REPETISJONSUKE

Alt er tilgjengelig for repetisjon, og det som ikke er mestret kan vektlegges 3 x så mye som resten.

MULIG EASTER EGG

Samle skall (shells) på noen fysiske steder og ved å gjøre spesifikke svar komboer som kan hjelpe med å ta vekk et feil alternativ ved flervalgspørsmål. Det skal ikke være rikelig av dem. Det skal ikke være mulig å finne et skall på samme måte eller sted flere ganger.

Se på hvor skall skal kunne samles etter hvert.

Appendix B

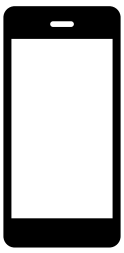
Question for week 1

UKE 1

PYTHON APP

MAY-LILL BAGGE OG

SOLVEIG RADDUM



UKE 1

Oppsumering av spørsmålgruppene

Spmørsmålstype	Kategori	# spørsmål pr leksjon	Mestring	Repetisjon	# spørsmål av denne typen
1.1 - Spørsmål	Diverse	1		3, 7, 10	43
1.2 - Matte operasjoner	Matte	2	Hver regel - 3 ganger rett og 2 rett på rad	4, 7, 10	13
1.3 - Typer	Typer	1	Hver regel - 10 ganger rett og 5 rett på rad	4, 7, 10	2
1.4 - Operator presendens	Matte	3	Hver regel - 3 ganger rett og 2 rett på rad	2, 7, 10	48
1.5 - Numerisk presisjon					
1.6 - Variabler og assignment	Variabler	2	Hver regel - 10 ganger rett og 5 rett på rad	3, 7, 10	11
1.7 - Augmented assignment	Variabler	2	Hver regel - 10 ganger rett og 5 rett på rad	4, 7, 10	14
1.8 - Feil	Feil	1	Hver regel - 10 ganger rett og 3 rett på rad	4, 7, 10	3

SPØRSMÅL 1.1 REPETISJON

1.1.1

Program

Presentasjon

Statement

TV sendeskjema

———————————-

Python

Matte oppgaver

Set med variabler

Set med instruksjoner

Kategori: Generelt

1.1.2

Programmeringsspråk

Fransk

Latin

Grammatikk

———————————-

Å si eller skrive ordet programmering

Variabler

Å taste inn noe på en datamaskin

Et kunstig språk som benyttes til å styre og kontrollere en datamaskin.

Kategori: Generelt

1.1.3

Paranteser

Krøllparanteser

Klammeparanteser

Spisse paranteser

———————————————-

( )

[ ]

{ }

< >

Kategori: Generelt

1.1.4

Paranteser

Krøllparanteser

Klammeparanteser

Spisse paranteser

———————————————

( )

[ ]

{}

< >

Kategori: Generelt

1.1.5

Paranteser

Krøllparanteser

Klammeparanteser

Spisse paranteser

———————————-

()

[]

{}

< >

Kategori: Generelt

1.1.6

Statement

Kommentar

Program

Uttrykk

———————————-

Partiprogram

Endring

Kommentar

Kommando som datamaskinen utfører

Kategori: Generelt

1.1.7

Prosessor

Hard drive

Main metode

Operativ system

———————————

Hovedregneenheten i en datamaskin som utfører intruksjonene gitt i et dataprogram.

Skjerm

Lagringsmedium for binært kodet informasjon for datamaskiner.

En kombinasjon av verdier, konstanter, variabler, operatorer og funksjoner som programmeringsspråket tolker og evaluerer og returnerer en annen verdi.

Kategori: Generelt

1.1.8

Hard drive

Prosessor

Operativ system

Google

———————————-

Lagringsmedium for binært kodet informasjon for datamaskiner.

Hovedregneenheten i en datamaskin som utfører intruksjonene gitt i et dataprogram.

Datamaskinen.

Et program som snakker med operativ systemet.

Kategori: Generelt

1.1.9

Operativ system

Python

Siri

Tastaturet

———————————————-

Det eneste programmet som snakker direkte med hardweren til en datamaskin.

Samling av operatorene i et programmeringsspråk.

Objekt orientering.

Program som tolker python programmer slik at det kan kjøres på en datamaskin.

Kategori: Generelt

1.1.10

Interpreter.

Filsystem.

Database.

Operativ system.

———————————————

Kjører programmet og oversetter det til noe operativ systemet kan forstå.

Et sett med verdier og et sett med operasjoner som kan brukes på disse verdiene.

Et kunstig språk som benyttes til å styre og kontrollere datamaskiner.

Filsystem og databaser.

Kategori: Generelt

1.1.11

Python skall

Programmeringsspråk

Kalkulator

Kommentar

———————————————————

Et sted hvor du taster inn en og en Python kommando og får respons etter hver av dem.

Noe som er planlagt å utføre i Python.

Python kommentarer.

Alle tegn som normalt er usynlig, slik som space og return.

Kategori: Generelt

1.1.12

Aritmestiske opreatorer

Operatorer

Aritmetiske uttrykk

Typer

$a^2 + b^2 = c^2$

$$\frac{sinA}{a} = \frac{sinB}{b} = \frac{sinC}{c}$$

{ } [ ]

*+ - * / // % ** (korrekt)*

Kategori: Matte

1.1.13

Operander

3 og 4

int og float

x

Verdi(er) som blir brukt med operatorer.

Folk som gjøre mattematikk.

Operasanger(e).

Utvikler(e).

Kategori: Matte

1.1.14

Uttrykk

Operander

Python

Interpreter

———————————————-

En kombinasjon av verdier, konstanter, variabler, operatorer og funksjoner som programmeringsspråket tolker og evaluerer og returnerer en annen verdi.

Et sett med verdier og et sett med operasjoner som kan brukes på disse verdiene.

Et kunstig språk som benyttes til å styre og kontrollere en datamaskin.

Et sted hvor du taster inn en og en Python kommando og får respons etter hver av dem.

Kategori: Generelt

1.1.15

Type

Heltall

Int

Uttrykk

_____

Et sett med verdier og et sett med operasjoner som kan brukes på disse verdiene.

Tall med komma

Heltall

Alle tegn som normalt er usynlig, slik som space og return.

Kategori: Typer

1.1.16

Int

Float

Utrykk

Tall med komma

_____-

Heltall

Tall med komma

Uttrykk

Typer

Kategori: Typer

1.1.17

Float

Uttrykk

Python

Int

———————————————-

Tall med komma

Heltall

Modulo

Unær operator

Kategori: Typer

1.1.18

Modulo

Floor

Type

Operander

———————————————————-

Resten etter en deling.

Deling av to heltall

Tall med komma

Type

Kategori: Matte

1.1.19

Intiger division

Tall med komma

Type

Modulo

———————————————————

Deling av to heltall

Resten etter en deling

Type

Å gi en variabel en verdi

Kategori: Matte

1.1.20

Floor

Minimum

Maximum

Float

_____

Det største heltallet mindre eller lik input tallet.

Bibliotek

Modul

Heltall biten til et float tall.

Kategori: Matte

1.1.21

Binære operatorer

Python

Minne

Unær operator

_____-

Operator som tar to operander.

Operator som funker på to typer

Kommentar

Alle operatorene i Python

Kategori: Matte

1.1.22

Unære operatorer

Binære operatorer

Overloaded operatorer

Alle operatorerne i Python

———————————————-

Operatorer som tar en operand

Operatorer som tar to operander

Overloaded operatorer

Alle operatorene i Python

Kategori: Matte

1.1.23

Overloaded operatoer

Binære operatorer

Unære operatoerer

Variabler

Operatorer som kan utføres på flere typer.

Variabler

Operator presedens

Å gi en variabel en verdi

Kategori: Typer

1.1.24

Operator presedens

Variabel

Unære operatorer, binære operatorer og overloaded operatorer

Binære operatoerer, unære operatoerer og overloaded operatoerer

---

Evalueringsrekkefølgen til matematiske operatorer.

Et navn som referrerer til en verdi

En forkortelse hvor en operator og assignment slåes sammen

Binære operatorer, unære operatoerer, overloaded operatorer

Kategori: Matte

1.1.25

Variabel

x

x = 4

Minne adresse

Et navn som refererer til en verdi

Minne addresse

Assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.26

Assignment statement

Variabel

Forkortelse hvor en operator og en assignment slåes sammen.

Augmented assignment

Å gi en variabel en verdi

Operator presedens

Unær operator

Minneadresse

Kategori: Variabler

1.1.27

Minne adresse

Variabel

Unær operator

Uttrykk

En adresse for hvert sted i datamaskinens minne.

Augmented assignment

Assignment statement

id1

Kategori: Generelt

1.1.28

Augmented assignemnt

id

Variabel

Operand

————————————————-

Forkortelse hvor en operator og en assignment slåes sammen.

Operand

En verdi som refererer til et navn

Operator

Kategori: Variabler

1.1.29

+= *(rett svar)*

Verdi

Variabel

++

————————————————-

Forkortelse for pluss og assignment.

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.30

-= (rett svar)

Verdi

Variabel

- -

_____

Forkortelse for minus og assignment.

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.31

**= (rett svar)*

Verdi

Variabel

**

———————————————-

Forkortelse for gange og assignment.

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.32

/= (rett svar)

Verdi

Variabel

//

———————————————

Forkortelse for deling og assignment.

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.33

//= (rett svar)

Verdi

Variabel

==

————————————————————-

Forkortelse for integer deling og assignment

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.34

%= (rett svar)

Verdi

Variabel

%%

————————————————————

Forkortelse for remaninder og assignment

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.35

**=

Operand

Operator assignment

En verdi som refererer til et navn

———————————————————-

Forkortelse for exponantiation og assignment

Operand

Operator assignment

En verdi som refererer til et navn

Kategori: Variabler

1.1.36

Syntax feil

Semantisk feil

Error

Operator feil

———————————-

En feil pga noe som ikke er gyldig Python kode.

En feil pga noe som ikke lar seg utføre

Error statement

Operator feil

Kategori: Feil

1.1.37

Semantisk feil

Syntax feil

Error statement

Operator feil

———————————

En feil pga noe som ikke lar seg utføre.

En feil pga noe som ikke lar seg utføre

Operator feil

Error

Kategori: Feil

1.1.38

Literal

Variabel

Bokstav

Navn

En konkret, fast verdi.

Bokstav

Variabel

Assignment

Kategori: Generelt

1.1.39

Anbefalt max linjelengde i Python

Max lengde et uttrykk kan ha i Python

Max variabel lengde i Python

Max whitespacei et program i Python

---

80 tegn

70 tegn

60 tegn

50 tegn

Kategori: Generelt

1.1.40

Whitespace

Backslash i Python

#

x = x * x

_____

Alle tegn som normalt er usynlig, slik som space og return.

#

Max linjelendge i Python

Python skall

Kategori: Generelt

1.1.41

Backslash

cont.

#

?

_____

Sier at koden fortssetter på linjen under og ikke tolk koden før den også er lest.

Deletegn

Begynnelsen på en kommentar

Forkortelse for deling og assignment

Kategori: Generelt

1.1.42

 # (rett svar)

?

>>>

\

_____-

Begynner en kommentar

Prompt

Linjefortsettelse

Help

Kategori: Kommentarer

1.1.43

Kommentar

Oppgave

Variabel

Assignment

————————————-

Forklaringer på hva kode gjør og hvordan.

Snakkingen man gjør mens man programmerer.

Minneaddresse

Oppgavetekst

Kategori: Kommentarer

1.2 MATTEOPERASJONER

Kategori: Matte - alle spm i 1.2

1.2.1

tall // tall (tall = int [1 til 50])

svaralternativer:

rett svar, tall 1 /tall 2, tall 2 // tall 1, tall 1 % tall 2

1.2.2

tall a // tall b (tall a = int [-49 til -1] tall b = int [1 til 50])

svaralternativer:

rett svar, -rett svar, rett svar +1, -(rett svar +1)

1.2.3

tall a // tall b (tall a = int [1 til 50] tall b = int [-49 til -1])

svaralternativer:

rett svar, -rett svar, rett svar +1, -(rett svar +1)

1.2.4

tall // tall (tall = int [-49 til -1])

svaralternativer:

rett svar, -rett svar, rett svar +1, -(rett svar +1)

1.2.5

tall % tall (tall = int[1 til 50])

svaralternativer:

rett svar, tall 1 / tall 2, -rett svar, tall 1 // tall 2

1.2.6

tall a % tall b (tall = int [1 til 50] tall b = int [-49 til -1])

svaralternativer:

rett svar, tall 1 / tall 2, -rett svar, rett svar +1

1.2.7

tall a % tall b (tall = int [-49 til -1] tall b = int [1 til 50])

svaralternativer:

rett svar, tall 1 / tall 2, -rett svar, rett svar +1

1.2.8

tall % tall (tall = int [-49 til -1])

svaralternativer:

rett svar, -rett svar, rett svar +1, -(rett svar +1)

1.2.9

tall ** tall (tall = int [-4 til -1 og int [1 til 4])

svaralternativer:

rett svar, -rett svar, tall 1 * tall 2, tall 1/tall 2

1.2.10

floor av tall a.tall b (tall a og b = int [0 til 9]

svaralternativ:

rett svar (tall a), rett svar +1, tall a.tall b, tall a**tall b

1.2.11

floor av tall a.tall b (tall a og b = int [-9 til 0])

svaralternativ:

rett svar (tall a), rett svar +1, tall a.tall b, tall a -1

1.2.12

int (tall a.tall b) (tall a og b = int [0 til 9])

svaralternativ:

rett svar (tall a), rett svar +1, tall a.tall b, tall a -1

1.2.13

int (a.tall.tall b) (tall a og b = int [-9 til 0])

svaralternativ:

rett svar (tall a), rett svar +1, tall a.tall b, tall a -1

1.3 TYPER

Kategori: Typer - alle spm i 1.3

1.3.1

Hvilke type er [int = [-50 til 100]]?

Svaralternativer:

Rett svar (Int), Float, Variable, Number

1.3.2

Hvilke type er [float = [-50 til 100]]?

Svaralternativer:

Rett svar (Float), Int, Variable, Number

1.4 OPERATOR PRESEDENS

Kategori: Matte - alle spm i 1.4

x = int [-12 til 12] i alle spm under, unatt når x skal være under brøkstrek, da kan det ikke være 0. I hvert spørsmål refererer x1 til samme tilfeldige tall osb. (Det samme gjelder for float)

float med et desimaltall, noen av svaralternativene kan være like til det rette svaret om de er en float med et eller annet .0

Alle svaralternativene kan tilfeldigvis bli lik det rette svaret om de tilfeldige tallene er er slik. Bør ha sjekk for det, og bare dublisere en av de andre alternativene om det skulle være tilfelle. Om alt annet feiler, bare velge noen tilfedige tall som er ulik det rette, men mellom -20 og +20. Float eller int alt etter hva som passer.

1.4.1

x1 + x2 - x3

Svaralternativer:

rett svar, -rett svar, x1 + (x2 - x3), x1 + x2 + x3

1.4.2

x1 + (x2 - x3)

Svaralternativer:

rett svar, -rett svar, x1 + x2 - x3, x1 + x2 + x3

1.4.3

(x1 + x2) - x3

Svaralternativer:

rett svar, -rett svar, x1 + (x2 - x3), x1 + x2 + x3

1.4.4

x1 * x2 / x3

Svaralternativer:

rett svar,, x1 + (x2 / x3), x1 * (x2 / x3), x1 / x2 * x3

1.4.5

x1 * (x2 / x3)

Svaralternativer:

rett svar,, (x1 + x2) / x3, x1 * x2 * x3, x1 / x2 / x3

1.4.6

(x1 * x2) / x3

Svaralternativer:

rett svar,, x1 + x2 / x3, x1 * x2 * x3, x1 / x2 / x3

1.4.7

(float1 // float2) / float3

Svaralternativer:

rett svar, float1 // float2 / float3, float1 / float2 / float3, float1 / float2 // float3

1.4.8

float1 // float2 / float3

Svaralternativer:

rett svar, float1 // (float2 / float3), float1 / float2 / float3, float1 / float2 // float3

1.4.9

float1 // (float2 / float3)

Svaralternativer:

rett svar, (float1 // float2) / float3, float1 / float2 / float3, float1 / float2 // float3

1.4.10

(float1 / float2) // float3

Svaralternativer:

rett svar, float1 // float2 / float3, float1 / float2 / float3, float1 // float2 // float3

1.4.11

float1 / float2 // float3

Svaralternativer:

rett svar, float1 / float2 / float3, float1 / float2 * float3, (float1 / float2) // float3

1.4.12

float1 / (float2 // float3)

Svaralternativer:

rett svar, (float1 // float2) / float3, float1 / float2 / float3, float1 / float2 ** float3

1.4.13

(x1 % x2) / x3

Svaralternativer:

rett svar, x1 % (x2 /x3), x1 % x2 % x3, x2 % x1 / x3

1.4.14

x1 % x2 / x3

Svaralternativer:

rett svar, x1 % (x2 /x3), x1 % x2 % x3, x2 % x1 / x3

1.4.15

x1 % (x2 / x3)

Svaralternativer:

rett svar, x1 % x2 /x3, x1 % x2 % x3, x2 % x1 / x3

1.4.16

(x1 / x2) % x3

Svaralternativer:

rett svar, x1 / (x2 % x3), x1 % x2 / x3, x2 % x1 / x3

1.4.17

x1 / x2 % x3

Svaralternativer:

rett svar, (x1 / x2) % x3, x1 % x2 % x3, x2 / x1 % x3

1.4.18

x1 / (x2 % x3)

Svaralternativer:

rett svar, (x1 / x2) % x3, x1 % x2 % x3, x2 % x1 / x3

1.4.19

x1 ** x2

Svaralternativer:

rett svar, x1 * x2, x2, x2 ** x1

1.4.20

x1 + x2 * x3

Svaralternativer:

rett svar, (x1 + x2) * x3, (x1 * x3) - x2, (x1 * x2) + x3

1.4.21

(x1 + x2) * x3

Svaralternativer:

rett svar, (x2 * x3) + x1, (x1 * x3) + x2, x3**3

1.4.22

x1 + (x2 * x3)

Svaralternativer:

rett svar, (x1 + x3) * x2, (x1 + x2) * x3, (x1 * x2) + x3

1.4.23

x1 - x2 * x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 - x2) * x3, x1 - x3 * x2

1.4.24

(x1 - x2) * x3

Svaralternativer:

rett svar, x1 + x2 * x3, x1 - x2 * x3, x1 - x3 * x2

1.4.25

x1 - (x2 * x3)

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 - x2) * x3, x1 - x3 * x2

1.4.26

x1 + x2 / x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 + x2) / x3, x1 + x3 / x2

1.4.27

(x1 + x2) / x3

Svaralternativer:

rett svar, x1 + x2 * x3, x1 + (x2 / x3), x1 + x3 / x2

1.4.28

x1 + (x2 / x3)

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 + x2) / x3, x1 + (x3 / x2)

1.4.29

x1 - x2 / x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 - x2) / x3, x1 - x3 / x2

1.4.30

(x1 - x2) / x3

Svaralternativer:

rett svar, x1 + x2 * x3, x1 - x2 / x3, x1 - x3 / x2

1.4.31

x1 - (x2 / x3)

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 - x2) / x3, x1 - x3 / x2

1.4.32

x1 * x2 + x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 + x2) / x3, x1 * (x3 + x2)

1.4.33

(x1 * x2) + x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 + x2) * x3, x1 + x3 * x2

1.4.34

x1 * (x2 + x3)

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 - x2) * x3, x1 + x3 * x2

1.4.35

x1 * x2 - x3

Svaralternativer:

rett svar, x1 - x2 * x3, (x1 - x2) / x3, x1 - x3 * x2

1.4.36

(x1 * x2) - x3

Svaralternativer:

rett svar, x1 * x2 / x3, (x1 + x2) - x3, x1 - x3 * x2

1.4.37

x1 * (x2 - x3)

Svaralternativer:

rett svar, x1 - x2 * x3, (x1 - x2) - x3, x1 - x3 * x2

1.4.38

x1 / x2 + x3

Svaralternativer:

rett svar, x1 + x2 * x3, (x1 + x2) / x3, x1 ** x2

1.4.39

(x1 / x2) + x3

Svaralternativer:

rett svar, x1 + x2 / x3, (x1 + x2) / x3, x1 + x3 / x2

1.4.40

x1 / (x2 + x3)

Svaralternativer:

rett svar, x1 / x2 * x3, (x1 * x2) / x3, x1 - x3 / x2

1.4.41

x1 / x2 - x3

Svaralternativer:

rett svar, x1 + x2 - x3, (x1 - x2) / x3, x1 - x3 / x2

1.4.42

(x1 / x2) - x3

Svaralternativer:

rett svar, x1 - x2 * x3, (x1 - x2) / x3, x1 - x3 / x2

1.4.43

x1 / (x2 - x3)

Svaralternativer:

rett svar, x1 + x2 / x3, (x1 - x2) / x3, x1 - x3 / x2

1.4.44

x1 ** x2 * x3

Svaralternativer:

rett svar, x1 ** (x2 * x3), (x1 ** x2) / x3, x1 ** x3 * x2

1.4.45

x1 ** (x2 / x3)

Svaralternativer:

rett svar, x1 ** (x2 * x3), (x1 ** x2) / x3, x1 ** x3 / x2

1.4.46

float1 // float2 * float3

Svaralternativer:

rett svar, float1 * (float2 // float3), (float1 * float2) // float3, float1 * float3 / float2

1.4.47

float1 // (float2 / float3)

Svaralternativer:

rett svar, float1 / (float2 // float3), (float1 / float2) // float3, float1 // float3 / float2

1.4.48

Hvilke rekkefølge er det best å addere disse tallene i Python?

float1 + float2 + float3 + float4

Svaralternativer:

rett svar (minst til størst), størst til minst, tilfeldig rekkefølge, rekkefløge fra spørsmålet

1.6 VARIABLER OG ASSIGNEMENT

Kategori: Variabler - alle spm i 1.6

1.6.1

x = tall

x = ?

Svaralternativer:

tall, tall +1, tall -1, Int

1.6.2

x = tall

x = ?

Svaralternativer:

tall, tall +1, tall -1, Float

1.6.3

x = tall + 3.0

x = ?

Svaralternativer:

tall + 3.0, tall +1.0, tall, Float

1.6.4

x = tall + 3

x = ?

Svaralternativer:

tall, tall +3, tall -1, Int

1.6.5

x = tall - 3

x = ?

Svaralternativer:

tall, tall -3, tall +1, Int

1.6.6

x = tall1

y = x * 3

y = ?

Svaralternativer:

rett svar, tall1, tall1/3, Int

1.6.7

x = tall1

y = x + 3

y = ?

Svaralternativer:

rett svar, tall1, tall1/3, Int

1.6.8

x = tall1

x = x * x

x = ?

Svaralternativer:

rett svar, tall1, tall1/3, Int

1.6.8

x = tall1

x = x + x

x = ?

Svaralternativer:

rett svar, tall1, tall1 * 3, Int

1.6.9

x = tall1

x = x**x

x = ?

Svaralternativer:

rett svar, tall1 * tall1, tall1, Int

1.6.10

x = tall1

x = x * x

x = ?

Svaralternativer:

rett svar, tall1, tall1/3, Int

1.6.11

x = tall1

x = x - x

x = ?

Svaralternativer:

rett svar, tall1, tall1/3, Int

1.7 AUGMENTED ASSIGNMENT

Kategori: Variabler - alle spm i 1.7

1.7.1

x = x1

x += x2

x = ?

Svaralternativer:

rett svar, x1, x2, Int

1.7.2

x = float1

x += float2

x = ?

Svaralternativer:

rett svar, float1, float2, Int

1.7.3

x = x1

x -= x2

x = ?

Svaralternativer:

rett svar, x1, x2, Int

1.7.4

x = float1

x -= float2

x = ?

Svaralternativer:

rett svar, float1, float2, -float1

1.7.5

x = x1

x *= x2

x = ?

Svaralternativer:

rett svar, x1, x2, x1/x2

1.7.6

x = float1

x *= float2

x = ?

Svaralternativer:

rett svar, float1, float2, Float

1.7.7

x = x1

x /= x2

x = ?

Svaralternativer:

rett svar, x1, x2, x1/x1

1.7.8

x = float1

x /= float2

x = ?

Svaralternativer:

rett svar, float1, float2, float1/float1

1.7.9

x = x1

x //= x2

x = ?

Svaralternativer:

rett svar, x1, x2, SyntaxError: invalid syntax

1.7.10

x = float1

x //= float2

x = ?

Svaralternativer:

rett svar, float1, float2, SyntaxError: invalid syntax

1.7.11

x = x1

x %= x2

x = ?

Svaralternativer:

rett svar, x1, x2, SyntaxError: invalid syntax

1.7.12

x = float1

x %= float2

x = ?

Svaralternativer:

rett svar, float1, float2, float1/float2

1.7.13

x = x1

x **= x2

x = ?

Svaralternativer:

rett svar, x1, x2, x1 * x2

1.7.14

x = float1

x **= float2

x = ?

Svaralternativer:

rett svar, float1, float2, float2**float1

1.7.15

x = float1

x = int(x)

x = ?

Svaralternativer:

rett svar, float1, Int, SyntaxError: invalid syntax

1.8 FEIL

Kategori: Feil - alle spm i 1.8

1.8.1

Hva menes med denne feilmeldingen?

```
print('Helo world)
                  ^
SyntaxError: EOL while scanning string literal
```

Svaralternativer:

Mangler utropstegn, feil type quote mark, feilstavet Helo, mangler quote etter world

1.8.2

Hva menes med denne feilmeldingen?

```
print(Helo world')
      ^
SyntaxError: invalid syntax
```

Svaralternativer:

Mangler utropstegn, feil type quote mark, feilstavet Helo, mangler quote før Helo

1.8.3

Hva menes med denne feilmeldingen?

```
print'Helo world')
      ^
SyntaxError: invalid syntax
```

Svaralternativer:

Mangler utropstegn, feil type quote mark, feilstavet Helo, mangler parantesbegynnelse før ‘

Appendix C

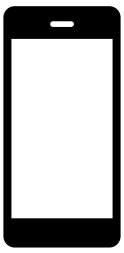
Questions used in week 2

UKE 3

PYTHON APP

MAY-LILL BAGGE OG

SOLVEIG RADDUM



UKE 3

Oppsumering av spørsmålsgruppene

Spmørsmålstype	Kategori	# spørsmål pr leksjon	Mestring	Repetisjon	# spørsmål
3.1 - Spørsmål	Diverse	2		6, 9, 12	4
3.2 - Tester	Test	7	Hver regel - 5 ganger rett og 3 rett på rad	6, 9, 12	4
3.3 - Feilmeldinger					
3.4 - Repetisjon	Diverse	2	Repetisjon denne uken: 1.1 og 2.3		

SPØRSMÅL 3.1 REPETISJON

Spørsmål av type A.

Alle spørsmål i denne uken (3.1 og 3.2) er av kategori test.

3.1.1

Quality assurance

Å ikke få feilmeldinger

Bugfixing

Unittesting

Undersøke at software fungerer rett.

Programmører som spiser Quality Street mens de programmerer.

Vanlig type software insureance.

Testing.

3.1.2

unit test

system test

bug fixing

assertEqual

Test av en isolert komponent av et program

Test av programmer

Test mange selskaper holder av programmører før de ansetter dem

Styrketest for datamaskiner

3.1.3

assertEqual(x,y)

TRUE

FALSE

assertUnequal(x,y)

———————————————————-

Funksjon i unittest som undersøker om de to første parameterne er like.

Funksjon som returnerer to like integers.

Funksjon som returernerer et antall integers som er likt til prarameterne.

Funksjon som undersøker om to programmører sitter på samme datamaskin.

3.1.4

system test

assertEqual

doctest

unit test

———————————————————-

Test av software som en helhet.

Systematisk testing.

Systemer som tester software.

Test av seriekoblede brannvarslsere.

SPØRSMÅL 3.2 TESTER

Steder med x1 til xn er tilfeldig int mellom -9 og 9. a og b er literaly med i spørsmålene, mens x'ene er der som det tilfeldige tallet, eventuelt med artimetiske operasjoner utført på dem. Spørsmålene kan bare stilles en vei.

3.2.1

Hvilke funksjonskropper ville bestått unittesten:

```
expected = (x1+x2)
actual = sum(x1, x2)
self.assertEquals(expected,actual, "The numbers are two
integers.")
```

Svaralternativer:

```
return (x1+x2)
return a+b
return (x1)+b
```

alle de overnevnte

3.2.2

Hvilke funksjonskropper ville bestått unittesten:

```
expected = (x1-x2)
actual = diff(x1, x2)
self.assertEquals(expected,actual, "The numbers are two
integers.")
```

Svaralternativer:

```
return (x1-x2)
return a-b
return (x1)-b
```

alle de overnevnte

3.2.3

Hvilke funksjonskropper ville bestått unittesten:

```
expected = (x1+x2)
actual = sum(x1, x2)
self.assertEquals(expected,actual, “The numbers are two
integers.”)
```

Svaralternativer:

```
return (x1-x2)
return a-b
return (x1)+b
```
alle de overnevnte

3.2.4

Hvilke funksjonskropper ville bestått unittesten:

```
expected = (x1-x2)
actual = diff(x1, x2)
self.assertEquals(expected,actual, “The numbers are two
integers.”)
```

Svaralternativer:

```
return (x1+x2)
return a+b
return (x1)-b
```
alle de overnevnte

Appendix D

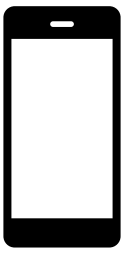
Questions used in week 2

UKE 4

PYTHON APP

MAY-LILL BAGGE OG

SOLVEIG RADDUM



UKE 4

Oppsumering av spørsmålsgruppene

Spmørsmålstype	Kategori	# spørsmål pr leksjon	Mestring	Repetisjon	# spørsmål
4.1 - Spørsmål	Diverse	2		5, 8, 11	
4.2 - Tester					0
4.3 - Strengoperasjoner	Strenger	3	Hver regel - 10 ganger rett og 5 rett på rad	5, 8, 11	
4.4 - Printe	Print	3	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11	
4.5 - Tastatur	Strenger	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11	
4.6 - Feilmeldinger	Feil	1	Hver regel - 10 ganger rett og 3 rett på rad	5, 8, 11	
4.7 - Repetisjon	Diverse	2	Repetisjon denne uken: 1.2, 1.3, 1.5, 1.7, 1.8, 2.2		

SPØRSMÅL 4.1 REPETISJON

4.1.1

String

_____-

En type på linje med int og float.

4.1.2

Character

_____

hvert element i en streng

4.1.3

str()

_____

typen til streng - casting til streng

4.1.4

int()

_____-

typen til heltall - casting til int

4.1.5

float()

_____-

typen til tall med komma - casting til float

4.1.6

‘

_____-

kan begynne og avslutte strenger

4.1.7

“

———————————————-

Kan begynne og avslutte strenger

4.1.8

opening and closing quotes must match

———————————————

4.1.9

EOL

———————————————

end of line

4.1.10

empty string

———————————————-

‘’

4.1.11

len

———————————————

innebygd funksjon som gir lengden til en streng

4.1.12

concatenation operator

————————————————-

+ på strenger

4.1.13

*

————————————————-

mulitplikasjon av strenger

4.1.14

escape character

4.1.15

escape sequence

4.1.16

\’

4.1.17

\”

4.1.18

\\

4.1.19

\t

4.1.20

\n

4.1.21

\r

4.1.22

print

4.1.23

valueError

4.1.24

kwargs

4.1.25

input

4.1.26

ValueError

SPØRSMÅL 4.3 STRENGOPERASJON

4.3.1

len

(length of escape sequence)

+

int

float

str

*

streng variabler

streng spanning multiple lines

SPØRSMÅL 4.4 PRINT()

4.4.1

print(1+1)

print('En streng')

print med \t

print multiline string

print int

print uten argumenter

print med argumenter av forskjellige typer

print med sep argument

print med end argument

SPØRSMÅL 4.5 TASTATUR

4.5.1

input

type

int(input())

input med prompt

SPØRSMÅL 4.6 FEILMELDINGER

4.6.1

```
'Charles Darwin"
                ^
SyntaxError: EOL while scanning string literal
```

```
'NH' + 3
```

```
TypeError: cannot concatenate 'str' and 'int' objects
```

```
9 + ' planets'
```

```
TypeError: unsupported operand type(s) for +: 'int' and ‘str’
```

```
int('a')

ValueError: invalid literal for int() with base 10: ‘a'


float('a')

ValueError: could not convert string to float: a


'That's not going to work'
      ^
SyntaxError: invalid syntax


'one
    ^
SyntaxError: EOL while scanning string literal
```

Appendix E

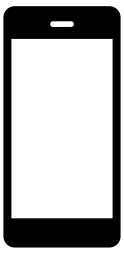
Questions for week 3

UKE 2

PYTHON APP

MAY-LILL BAGGE OG

SOLVEIG RADDUM



UKE 2

Oppsumering av spørsmålsgruppene

Spmørsmålstype	Kategori	# spørsmål pr lekson	Mestring	Repetisjon	# spørsmål
2.1 - Spørsmål	Diverse	1		5, 8, 11	27
2.2 - Innebygde funksjoner	Diverse	1	Hver regel - 5 ganger rett og 3 rett på rad	4, 8, 11	15
2.3 - Funksjoner	Diverse	3	Hver regel - 10 ganger rett og 5 rett på rad	3, 8, 11	61
2.4 - Feilmeldinger	Feil	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11	8
2.5 - Kommentarer	Kommentarer	1	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11	0
2.6 - Tester	Test	2	Hver regel - 10 ganger rett og 3 rett på rad	5, 8, 11	0
2.7 - Navn	Variabler	2	Hver regel - 5 ganger rett og 3 rett på rad	5, 8, 11	16
2.8 - Repetisjon	Diverse	1	Repetisjon denne uken: 1.4		

SPØRSMÅL 2.1 REPETISJON

2.1.2

Built in function

function call

string

scope

———————————————-

funksjoner som kommer med Python

function call

local variable

variable

2.1.3

abs

pow

absolutt

help(absolutt)

———————————————-

Innebygd Python funksjon som returnerer den absolutte verdien til sitt parameter.

Python sine kraftigste metoder som er som dens abdominal muscles

Innebygd funksjon i Python som setter en variabel til sin absolutte verdi

Verdier som sendes til funksjoner

2.1.4

function call

function definition

function

function namespace

_____-

<<function name>>(<<arguments>>)

<<function name>>

help(<<function name>>)

<<argument>>.<<function name>>

2.1.5

argument

help

self

return

_____

Verdier sendt til funksjoner

abs

scope

id

2.1.6

pow

abs

return

diff

_____

Innebygd Python funksjon som returnerer det første parameteret opphøyd i det andre parameteret.

Innebygd Python funksjon som returnerer det andre parameteret opphøyd i det første parameteret.

Innebygd Python funksjon som returnerer differensen mellom de to parameterne.

Innebygd Python funksjon som returnerer den sammenlagte styrken til paramterne.

2.1.7

help

?

describe

docstring

———————————————————-

Innebygd Python funksjon som returnerer en beskrivelse av hva en funksjon gjør.

Innebygd Python funksjon som returnerer all kjent informasjon om variabler.

Innebygd Python funksjon som returnerer typen til en varabel.

Alle de overnevnte.

2.1.8

round

id

def

scope

———————————————————-

Innebygd Python funksjon som returnerer den avrundede verdien til sitt parameter.

Innebygd Python funksjon som returnerer det første parameteret opphøyd i det andre parameteret.

Innebygd Python funksjon som returnerer hvor stor del av en sirkel gradene i paramtereret er.

Innebygd Python funksjon som returnerer en runding.

2.1.9

id

return

<<variable name>>

def(<<variabel name>>)

_____

Innebygd Python funksjon som returnerer id'en til en variabel.

Innebygd Python funksjon som returnerer verdien til parameter(ne).

Innebygd Python funksjon som returnerer definisjonen til en annen (navngitt) funksjon.

Innebygd Python funksjon som setter id'en til en variabel.

2.1.10

return

result

x

None

_____-

Innebygd Python funksjon som returnerer verdien til resten av statementen.

Innebygd Python funksjon som returnerer verdien til parameter(ne).

Innebygd Python funksjon som returnerer definisjonen til en annen (navngitt) funksjon.

Innebygd Python funksjon som setter id'en til en variabel.

2.1.11

def

scope

var

return

———————————————————-

Begynnelsen på alle funksjonsdefinisjoner i Python.

Innebygd Python funksjon som returnerer definisjonen til en annen (navngitt) funksjon.

Parameterliste.

Begynnelsen på documentation til en funksjon.

2.1.12

parameter

scope

preconditions

docstring

———————————————-

Verdier sendt til en funksjon når den kalles.

Navn gitt til argumenter når de mottas av en funksjon.

Begynnelsen på alle funksjonsdefinisjoner i Python.

Innebygd Python funksjon som returnerer definisjonen til en annen (navngitt) funksjon.

2.1.13

local variables

bestemte variabler

lokasjonsvariabler

def

———————————————-

variabeler som er gyldige i en liten, bestemt del av et program

variabeler som er gyldig i et program men ikke moduler

variabler som bare refererer til et bestemt sted (lokasjon) i minnet

variabler som bare gjelder på den linjen de er skrevet

2.1.14

scope

definisjonsområde

namespace

minneområde

_____

området av et program hvor en variabel er gyldig

vaiabler

abs

stedet i minent hvor en variabel er satt

2.1.15

namespace

variabel liste

minneområde

ekvivalens relasjon

_____-

en gruppering av navn slik at du kan skille forskjellig bruk av det samme navnet

plassen et navn tar på skjermen

mengden minne satt av til et variabel eller funksjonsnavn

området av et program hvor en funksjon er definert

2.1.16

documentation

wikipedia

function header

paramenterne

---

beskrivelse av en programdel som mennesker er ment å lese

tekst fil(er)

ASCII tekst

navn gitt til argumenter når de mottas av en funksjon

2.1.17

docstring

parameter ærklæring

in-ut ærklæring

type dokument

———————————————————-

beskrivelse av en funksjon hvor type(ne) til paramterne og returverdien er gitt

dokumentering av typen string

Strenger hentet fra dokumenter/tekst filer

Strenger skrevet til dokumenter/tekst filer

2.1.18

Test

Funksjon

Eksempel

Code expectation

———————————————-

Kodekall som undersøker om man får tilbake det forventede av en annen kodebit

Streng som finner feil i kode

Feilmelding

TestError

2.1.19

None

Ø

Null

Tom fil

———————————————-

Fravær av verdi

Tom fil

Det samme som null

No open node exeption

2.1.20

Precondition

Feilmelding

TypeError

Code expectation

———————————————

Krav til input til en funksjon eller programbit

Krav til programmører

Moduler som trenger bearbeiding for å passe inn i programmer

Funksjoner som trenger trening for å fungere i programmer

2.1.21

typekontrakt

funksjons operasjon

retur type

rette feil

_____-

kontrakt med en funksjon som sier at om du gir de(n) rette typen parametere vil du få den rette typen returnert

kontrakt med en ung mann (en type)

feilmeldinger om typer

parameter liste

2.1.22

feilmelding

paramenterliste

midleritdige variabler som er kalt der de ikke er gyldige

None

_____

meldinger om feil i programmet ditt fra Python tolken

type retur

kode som ikke er indentert

scope error

2.1.23

NameError

IndentationError

TypeError

SyntaxError

_____

feilmelding du får når et navn i koden ikke refererer til noe i den kodebiten hvor den står

feilstavede navn

feil i navnevalg

feilmelding du får når Python tolken forventer indentert kode

2.1.24

IndentationError

Indentert kode som inneholder feil

TypeError

PythonError

———————————————-

feilmelding du får når Python tolken forventer indentert kode

en feil i indentert kode

det samme som FunctionError

Variabler som bare har verdi på den linjen den blir satt

2.1.25

TypeError

ArgumentError

ParamenterError

NameError

———————————————-

Feilmelding fra Python tolken du får om en funksjon er kalt med feil antall argumenter

Feil gjort av en ung mann (en type)

Feilmelding fra Python tolken du får om du bruker en midlertidig varabel utenfor dens scope

Returverdi av feil type

2.1.26

Temporary variables

Global variables

Integer variabler

Float variabler

Variabler som bare har verdi i et begrenset områdråde av et program

Variabler som har verdier som er temperatur/grader

Variabler med typen tmp

Midlertidige programmer

2.1.27

Returverdi

Funksjonsverdien

Funksjonsvariabelen

Def verdien

Verdien en funksjon returnerer

Variabler med typen retur

Verdien til tur x 2

Verdier som blir tatt vare på

SPØRSMÅL 2.2 INNEBYGDE FUNKSJONER

x = int[1 til 9] hvor x1 osb er det samme tallet i et spørsmål.

y = float[1 til 9] hvor y1 er det samme tallet i et spørsmål.

2.2.1

Hva returnerer abs(y)?

Hva returnerer pow(y)?

Hva returnerner round(y)?

Hva returnerer sum(y)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [-abs(y)]?

Hvilke kode vil gi [y]?

Hvilke kode vil gi [round(y)]?

Hvilke kode vil gi [rett svar]?

2.2.2

Hva returnerer abs(-y)?

Hva returnerer pow(y)?

Hva returnerner round(y)?

Hva returnerer sum(y)?

————————————————-

Svaralternativer:

Hvilke kode vil gi [abs(y+1)]?

Hvilke kode vil gi [y]?

Hvilke kode vil gi [round(y)]?

Hvilke kode vil gi [rett svar]?

2.2.3

Hva returnerer abs(x)?

Hva returnerer pow(x)?

Hva returnerner round(x)?

Hva returnerer sum(x)?

————————————————-

Svaralternativer:

Hvilke kode vil gi [-abs(x)]?

Hvilke kode vil gi [x]?

Hvilke kode vil gi [round(x)]?

Hvilke kode vil gi [rett svar]?

2.2.4

Hva returnerer abs(-x)?

Hva returnerer pow(x)?

Hva returnerner round(x)?

Hva returnerer sum(x)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [-abs(x)]?

Hvilke kode vil gi [x]?

Hvilke kode vil gi [round(x)]?

Hvilke kode vil gi [rett svar]?

2.2.5

Hva returnerer pow(x1, x2)?

Hva returnerer pow(x)?

Hva returnerner round(x)?

Hva returnerer sum(x1, x2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(x2, x1)]?

Hvilke kode vil gi [x1]?

Hvilke kode vil gi [x2]?

Hvilke kode vil gi [rett svar]?

2.2.6

Hva returnerer pow(x1, -x2)?

Hva returnerer pow(x1)?

Hva returnerner retrun x1?

Hva returnerer sum(x1, x2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(-x2, x1)]?

Hvilke kode vil gi [-x1]?

Hvilke kode vil gi [x2]?

Hvilke kode vil gi [rett svar]?

2.2.7

Hva returnerer pow(-x1, -x2)?

Hva returnerer pow(-x1)?

Hva returnerner return -x2?

Hva returnerer sum(-x1, -x2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(x2, -x1)]?

Hvilke kode vil gi [-x1]?

Hvilke kode vil gi [-x2]?

Hvilke kode vil gi [rett svar]?

2.2.8

Hva returnerer pow(y1, y2)?

Hva returnerer pow(-y1)?

Hva returnerner return -y2?

Hva returnerer sum(-y1, -y2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y2, -y1)]?

Hvilke kode vil gi [-y1]?

Hvilke kode vil gi [-y2]?

Hvilke kode vil gi [rett svar]?

2.2.9

Hva returnerer pow(y1, -y2)?

Hva returnerer pow(-y1)?

Hva returnerner return -y2?

Hva returnerer sum(-y1, -y2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y2, -y1)]?

Hvilke kode vil gi [-y1]?

Hvilke kode vil gi [-y2]?

Hvilke kode vil gi [rett svar]?

2.2.10

Hva returnerer pow(-y1, -y2)?

Hva returnerer pow(-y1)?

Hva returnerner return -y2?

Hva returnerer sum(-y1, -y2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y2, -y1)]?

Hvilke kode vil gi [-y1]?

Hvilke kode vil gi [-y2]?

Hvilke kode vil gi [rett svar]?

2.2.11

Hva returnerer round(y)?

Hva returnerer pow(y)?

Hva returnerner return -y?

Hva returnerer sum(y1, y2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y)]?

Hvilke kode vil gi [-y]?

Hvilke kode vil gi [-y+3]?

Hvilke kode vil gi [rett svar]?

2.2.12

Hva returnerer round(-y)?

Hva returnerer pow(-y)?

Hva returnerner return -y?

Hva returnerer sum(y1, y2)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y)]?

Hvilke kode vil gi [y]?

Hvilke kode vil gi [-y+3]?

Hvilke kode vil gi [rett svar]?

2.2.13

Hva returnerer round(pow(-y,(abs(-x))?

Hva returnerer pow(y)?

Hva returnerner return -y?

Hva returnerer sum(-y, -x)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(-y)]?

Hvilke kode vil gi [-y]?

Hvilke kode vil gi [-y+3]?

Hvilke kode vil gi [rett svar]?

2.2.14

Hva returnerer round(pow(y,(abs(x))?

Hva returnerer pow(-y)?

Hva returnerner return y?

Hva returnerer sum(y, x)?

———————————————-

Svaralternativer:

Hvilke kode vil gi [pow(y)]?

Hvilke kode vil gi [-y]?

Hvilke kode vil gi [-y+3]?

Hvilke kode vil gi [rett svar]?

2.2.15

Hva returnerer round(pow(-y,-x))?

Hva returnerer pow(-y)?

Hva returnerner return -y?

Hva returnerer sum(-y, -x)?

_____-

Svaralternativer:

Hvilke kode vil gi [pow(-y)]?

Hvilke kode vil gi [-y]?

Hvilke kode vil gi [-y+3]?

Hvilke kode vil gi [rett svar]?

SPØRSMÅL 2.3 FUNKSJONER

De første spørsmålene er hva mangler her - først kommer kode med noe som mangler, så kommer rett kode, og til slutt svaralternativene.

2.3.1

```
    add(x, y):
...     """Return x plus y"""
...     return x + y
```

_____-

```
def add(x, y):
...     """Return x plus y"""
...     return x + y
```

_____

Svaralternativer:

return

function

sum

def

2.3.2

```
    sum(x, y):
...     """Return x plus y"""
...     return x + y
```

```
def sum(x, y):
...     """Return x plus y"""
...     return x + y
```

Svaralternativer:

return

function

sum

def

2.3.3

```
    subtract(x, y):
...     """Return x minus y"""
...     return x - y
```

```
def subtract(x, y):
...     """Return x minus y"""
...     return x - y
```

Svaralternativer:

return

scope

minus

def

2.3.4

```
    mulitply(x, y):
...     """Return x times y"""
...     return x * y
```

```
def multiply(x, y):
...     """Return x times y"""
...     return x * y
```

Svaralternativer:

return

scope

function

def

2.3.5

```
    divide(x, y):
...     """Return x divided by y"""
...     return x / y
```

```
def divide(x, y):
...     """Return x divided by y"""
...     return x / y
```

Svaralternativer:

return

function

sum

def

2.3.6

```
def add(x, y):
...     """Return x plus y"""
...     [redacted]    x + y
```

```
def add(x, y):
...     """Return x plus y"""
...     return x + y
```

Svaralternativer:

return

scope

sum

add

2.3.7

```
def sum(x, y):
...     """Return x plus y"""
...     [redacted]    x + y
```

```
def sum(x, y):
...     """Return x plus y"""
...     return x + y
```

Svaralternativer:

return

scope

find

def

2.3.8

```
def subtract(x, y):
...     """Return x minus y"""
...         x - y
```

```
def subtract(x, y):
...     """Return x minus y"""
...     return x - y
```

Svaralternativer:

return

scope

fun

def

2.3.9

```
def multiply(x, y):
...     """Return x times y"""
...         x * y
```

```
def multiply(x, y):
...     """Return x times y"""
...     return x * y
```

Svaralternativer:

return

scope

show

def

2.3.10

```
def divide(x, y):
...     """Return x divided by y"""
...         x / y
```

```
def divide(x, y):
...     """Return x divided by y"""
...     return x / y
```

Svaralternativer:

return

fun

min

def

2.3.11

```
    convert_to_celcius(farenheit):
     return ((farenheit -32) * 5/9)
```

```
def convert_to_celcius(farenheit):
     return ((farenheit -32) * 5/9)
```

Svaralternativer

def

fun

open

beg

2.3.12

```
def convert_to_celcius(farenheit):
    ((farenheit -32) * 5/9)
```

```
def convert_to_celcius(farenheit):
     return ((farenheit -32) * 5/9)
```

Svaralternativer

def

fun

open

beg

2.3.13

```
    quadratic(a, b, c, x):
     return (a*x**2) + (b*x) + c
```

```
def quadratic(a, b, c, x):
     return (a*x**2) + (b*x) + c
```

Svaralternativer:

if

for

def

fun

2.3.14

```
def quadratic(a, b, c, x):
    (a*x**2) + (b*x) + c
```

```
def quadratic(a, b, c, x):
     return (a*x**2) + (b*x) + c
```

_____

Svaralternativer:

if

for

def

fun

2.3.15

```
    abs_diff(a, b):
     return abs(a-b)
```

_____-

```
def abs_diff(a, b):
     return abs(a-b)
```

_____-

Svaralternativer:

def

run

beg

for

2.3.16

```
def abs_diff(a, b):
    abs(a-b)
```

_____-

```
def abs_diff(a, b):
     return abs(a-b)
```

Svaralternativer:

def

run

beg

for

2.3.17

```
    km_to_miles(km):
     return km*1.6
```

```
def km_to_miles(km):
     return km*1.6
```

Svaralternativer:

run

return

begin

function

2.3.18

```
def km_to_miles(km):
    km*1.6
```

```
_____________________

def km_to_miles(km):
     return km*1.6
_____________________
```

Svaralternativer:

run

return

begin

function

2.3.19

```
    avrage_grades(one, two, three):
     return ((one+two+three)/3)
_____________________________________

def avrage_grades(one, two, three):
     return ((one+two+three)/3)
_____________________________________
```

Svaralternativer:

fun

run

def

sum

2.3.20

```
def avrage_grades(one, two, three):
    ((one+two+three)/3)
```

```
def avrage_grades(one, two, three):
     return ((one+two+three)/3)
```

Svaralternativer:

ening

return

sort

sum

2.3.21

```
 square(num):
     return (num*num)
```

```
def square(num):
     return (num*num)
```

Svaralternativer:

run

for

until

next

2.3.22

```
def square(num):
    ________(num*num)
```

_____

```
def square(num):
     return (num*num)
```

_____

Svaralternativer:

run

for

until

next

2.3.23

```
____ next(num):
     return (num+1)
```

_____

```
def next(num):
     return (num+1)
```

_____

Svaralternativer:

run

for

until

next

2.3.24

```
def next(num):
   [redacted] (num+1)
```

```
def next(num):
     return (num+1)
```

Svaralternativer:

run

for

until

next

Etter dette er spørsmålene er hva vil denne koden returnere. Først kommer noe kode, så kommer fire alternativer for hva den returnerer. Kan derfor bare spørres en vei.

num = tilfeldig int mellom 1 og 9 eller -9 og -1

num1 til num n er samme tall i samme kodesnutt og den svaralternativer (num brukes når det bare er et tall som går igjen i oppgaven).

sjekke at svaralternativene ikke er like, om de er like alternativ1 +1osb.

2.3.25

```
def add(x, y):
...     """Return x plus y"""
...     return x + y
add(num1, num2)
_____________
```

Svaralternativer:

num1+num2

num1+num2+1

num1*num2

TypeError

2.3.26

```
x = num1
y = num2
def add(x, y):
...     """Return x plus y"""
...     return x + y
add(x,y)
_____________
```

Svaralternativer:

num1+num2

num1+num2+1

num1*num2

TypeError

2.3.27

```
x = num1
y = num2
def add(x, y):
```

```
...     """Return x plus y"""
...     x = num3
...     return x + y
add(x, y)
```

Svaralternativer:

num1+num2

num1+num3

num1+num2+num3

TypeError

2.3.28

```
x = num1
y = num2
def add(x, y):
...     """Return x plus y"""
...     y = num3
...     return x + y
add(x, y)
```

Svaralternativer:

num1+num2

num1+num3

num1+num2+num3

TypeError

2.3.29

```
def sum(x, y):
...     """Return x plus y"""
...     return x + y
sum(num1, num2)
```

_____-

Svaralternativer:

num1+num2

num1+num2+1

num1*num2

TypeError

2.3.30

```
x = num1
y = num2
def sum(x, y):
...     """Return x plus y"""
...     return x + y
sum(x,y)
```

_____-

Svaralternativer:

num1+num2

num1+num2+1

num1*num2

TypeError

2.3.31

```
x = num1
y = num2
def sum(x, y):
...     """Return x plus y”""
...     x = num3
...     return x + y
sum(x, y)
```

_____-

Svaralternativer:

num1+num2

num1+num2+num3

num1+num3

TypeError

2.3.32

```
x = num1
y = num2
def sum(x, y):
...     """Return x plus y"""
...     y = num3
...     return x + y
sum(x, y)
```

Svaralternativer:

num1+num2

num1+num2+num3

num1+num2

TypeError

2.3.33

```
def subtract(x, y):
...     return x - y
subtract(num1, num2)
```

Svaralternativer:

num1-num2

num1-num2-1

num1+num2

TypeError

2.3.34

```
x = num1
y = num2
def subtract(x, y):
...     return x - y
subtract(x,y)
_____________
```

Svaralternativer:

num1-num2

num1-num2-1

TypeError

num1/num2

2.3.35

```
x = num1
y = num2
def subtract(x, y):
...     x = num3
...     return x - y
subtract(x, y)
_____________
```

Svaralternativer:

num1-num2

num1-num2-1

num1-num3

TypeError

2.3.36

```
x = num1
y = num2
def subtract(x, y):
...     y = num3
...     return x - y
subtract(x, y)
```

Svaralternativer:

num1-num2

TypeError

num1-num3

num1/num2

2.3.37

```
def multiply(x, y):
...     return x * y
multiply(num1, num2)
```

Svaralternativer:

num1+num2

num1+num2-2

num1*num2

TypeError

2.3.38

```
x = num1
y = num2
def multiply(x, y):
...     return x * y
multiply(x,y)
```

Svaralternativer:

num1+num2

TypeError

num1*num2

num1/num2

2.3.39

```
x = num1
y = num2
def multiply(x, y):
...     x = num3
...     return x * y
multiply(x, y)
```

Svaralternativer:

num2*num3

TypeError

num1*num2

num1*num3

2.3.40

```
x = num1
```

```
y = num2
def multiply(x, y):
...     y = num3
...     return x * y
multiply(x, y)
_____________
```

Svaralternativer:

num1*num2

num1+num2+1

num1*num3

num2*num3

2.3.41

```
def divide(x, y):
...     return x / y
divide(num1, num2)
_____________
```

Svaralternativer:

num2/num1

num1/num2+1

num1*num2

num1/num2

2.3.42

```
x = num1
y = num2
def divide(x, y):
...     return x / y
divide(x,y)
_____________
```

Svaralternativer:

num2/num1

num1/num2+1

TypeError

num1/num2

2.3.43

```
x = num1
y = num2
def divide(x, y):
...     x = num3
...     return x / y
divide(x, y)
```

Svaralternativer:

num1/num3

num3/num2

num2/num1

num1/num2

2.3.44

```
x = num1
y = num2
def divide(x, y):
...     y = num3
...     return x / y
divide(x, y)
```

Svaralternativer:

num1/num3

num3/num2

num2/num1

num1/num2

2.3.45

```
def square(num):
     return (num*num)
square(num)
```

Svaralternativer:

num*num

num/num

TypeError

num+num

2.3.46

```
x = num
def square(num):
     return (num*num)
square(x)
```

Svaralternativer:

num*num

TypeError

num*x

x*x

2.3.47

```
x = num1
def square(x):
     x = num2
     return (x*x)
square(x)
```

Svaralternativer:

num1*num1

num2*num2

num1*num2

num1+num2

2.3.48

```
y = num1
def square(x):
...      y = num2
...      return (x*x)
square(y)
```

Svaralternativer:

num1*num2

num1*num1

num2*num2

num1+num2

2.3.49

Spørsmålene under spør etter typekontrakten til funksjonene:

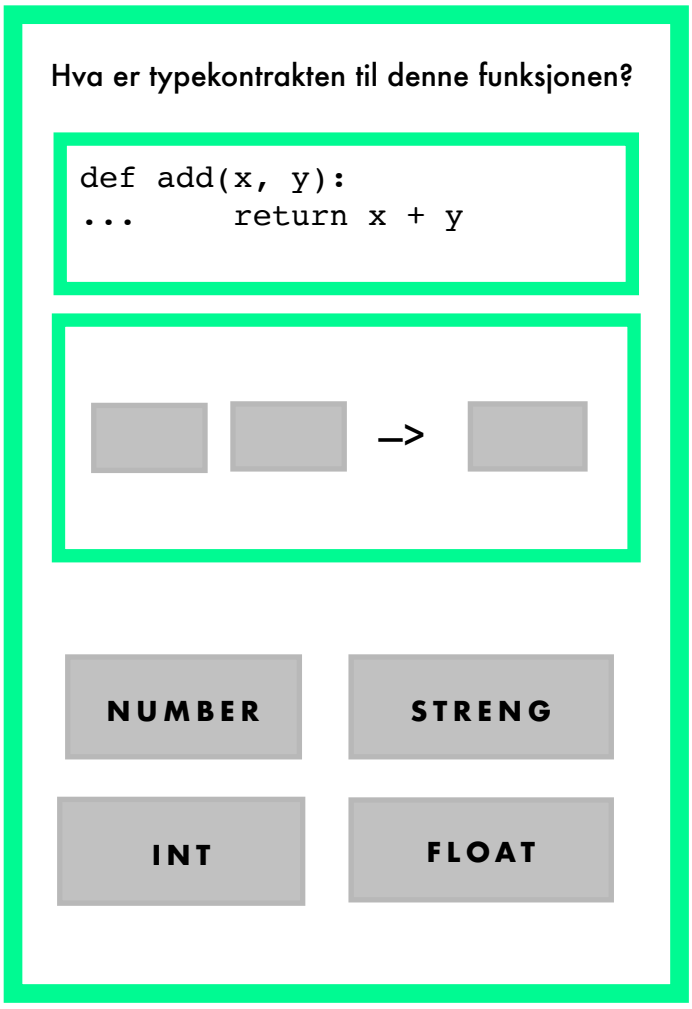
```
def add(x, y):
...     """Return x plus y"""
...     return x + y
```

Svarmuligheter

Her kan de velge fritt mellom typene vi møtt så langt:

Int, float, Streng, char og number, og skal enten dra dem på plass eller klikke på dem i rett rekkefølge (alt etter som hva som er lettest å implementere). Vil helst ha number når det ikke er noe spesielt som sier at det bør være int eller float.

Eksempelskjerm:



Svarmulighetene blir like for dem alle. (Bruker ikke char her.) Under funksjonen blir det rette svaret gitt.

number, number -> number

2.3.50

```
def sum(x, y):
...     return x + y
_____________________________-
```

Rett svar:

number, number -> number

2.3.51

```
def subtract(x, y):
...     return x - y
_________________________
```

Rett svar:

number, number -> number

2.3.52

```
def subtract(x, y):
...     return x - y
_________________________
```

Rett svar:

number, number -> number

2.3.53

```
def multiply(x, y):
...     return x * y
```

_____

Rett svar:

number, number -> number

2.3.54

```
def divide(x, y):
...     return x / y
```

_____

Rett svar:

number, number -> number

2.3.55

```
def convert_to_celcius(farenheit):
     return ((farenheit -32) * 5/9)
```

_____

Rett svar:

number -> number

2.3.56

```
def quadratic(a, b, c, x):
     return (a*x**2) + (b*x) + c
```

_____

Rett svar:

number, number, number, number -> number

2.3.57

```
def abs_diff(a, b):
     return abs(a-b)
```

Rett svar:

number, number -> int

2.3.58

```
def km_to_miles(km):
     return km*1.6
```

Rett svar:

number -> number

2.3.59

```
def avrage_grades(one, two, three):
     return ((one+two+three)/3)
```

Rett svar:

number, number, number -> number

2.3.60

```
def square(num):
     return (num*num)
```

Rett svar:

number -> number

2.3.61

```
def next(num):
     return (num+1)
```

Rett svar:

number -> number

SPØRSMÅL 2.4 FEILMELDINGER

Etter dette er spørsmålene er hva vil denne koden returnere. Først kommer noe kode, så kommer fire alternativer for hva den returnerer. Kan derfor bare spørres en vei.

num = tilfeldig int mellom 1 og 9 eller -9 og -1

num1 til num n er samme tall i samme kodesnutt og den svaralternativer (num brukes når det bare er et tall som går igjen i oppgaven).

sjekke at svaralternativene ikke er like, om de er like alternativ1 +1osb.

2.4.1

```
def add(x, y):
return x + y
add(num1, num2)
```

Svaralternativer:

num1+num2

num1+num2+1

IndentationError

x

2.4.2

```
def sum(x, y):
return x + y
sum(num1, num2)
```

Svaralternativer:

num1+num2

IndentationError

num1*num2

TypeError

2.4.3

```
x = num1
y = num2
def sum(x, y):
return x + y
sum(x,y)
```

Svaralternativer:

num1+num2

IndentationError

num1*num2

TypeError

2.4.4

```
x = num1
y = num2
def sum(x, y):
x = num3
...     return x + y
sum(x, y)
_____________
```

Svaralternativer:

num1+num2

IndentationError

num1+num3

TypeError

2.4.5

```
x = num1
y = num2
def sum(x, y):
y = num3
...     return x + y
sum(x, y)
_____________
```

Svaralternativer:

num1+num2

IndentationError

num1+num2

TypeError

2.4.6

```
def subtract(x, y):
return x - y
subtract(num1, num2)
```

———————-

Svaralternativer:

num1-num2

IndentationError

num1+num2

TypeError

2.4.7

```
x = num1
y = num2
def subtract(x, y):
return x - y
subtract(x,y)
```

———————-

Svaralternativer:

num1-num2

IndentationError

TypeError

num1/num2

2.4.8

```
def multiply(x, y):
return x * y
multiply(num1, num2)
```

———————-

Svaralternativer:

num1+num2

IndentaionError

num1*num2

TypeError

SPØRSMÅL 2.7 NAVN

For hvert spørsmål (de fleste) generers enten:

Hva vil du tro denne variabelen står for? eller

Hva vil være et godt variabel navn for dette?

(Denne variabelen er da ordet i italic over streken, og dette vil være ordet i italic under streken. De andre ordene blir alternativer.)

Her må de altid få alternativer. Ikke taste.

Kategori for alle er variabler.

2.7.1

grader_celcius

grader_farenheit

katet

first_value

————————————-

hypotenusen til en trekant

det er et teit navn som ingen vil huske hva står for om et halvt år

grader

grader i celsius

2.7.2

grader_celcius

grader_farenheit

grader

first_value

———————————-

hypotenusen til en trekant

grenseverdier

det er et teit navn som ingen vil huske hva står for om et halvt år

grader i farenheit

2.7.3

hypotenus

ene_siden

langsiden

katet

_____

det er et teit navn som ingen vil huske hva står for om et halvt år

hypotenus

katet

langsiden

2.7.4

katet1

student

side

en_kortside

_____-

det er et teit navn som ingen vil huske hva står for om et halvt år

det ene katetet

hypotenus

den øverste siden i en trekant

2.7.5

katet2

hypotenusen

oppsiden

flat_side_of_triangle

———————————-

det er et teit navn som ingen vil huske hva står for om et halvt år

det ene katetet

hypotenus

den øverste siden i en trekant

2.7.6

volum_sirkel

volum

sirkel

areal

———————————————

volumet av en sirkel

det er et teit navn som ingen vil huske hva står for om et halvt år

arealet til en sirkel

hypotenus

2.7.7

bok_tittel

tittel

bok

overskrift

————————————————-

tittelen på en bok

det er et teit navn som ingen vil huske hva står for om et halvt år

forfatteren til en bok

alt som står utenpå en bok

2.7.8

bok_forfatter

bok

forfatter

body

————————————————

det er et teit navn som ingen vil huske hva står for om et halvt år

forfatteren til en bok

tittelen på en bok

alt som står utenpå en bok

2.7.9

(kan bare spørres en vei)

bok_år

_____-

det er et teit navn som ingen vil huske hva står for om et halvt år

året en bok er utgitt

året en bok er trykt

året en bok ble anskaffet

2.7.10

student

spiller

deltager

person

_____

en student

en deltager

en spiller

en forleser

2.7.11

student

spiller

deltager

foreleser

_____

en student

en deltager

en spiller

en forleser

2.7.12

uke

syv_dager

rotasjon

periode

_____

en uke

en periode

en runde

en måned

2.7.13

(kan bare spørres en vei)

ukedag

_____-

det er et teit navn som ingen vil huske hva står for om et halvt år

hverdager

hverdager + helger

en hvilken som helst dag i uken

2.7.14

(kan bare spørres en vei)

grader

_____

grader celcius

grader farenheit

det er et teit navn som ingen vil huske hva står for om et halvt år

1/360 del av en cirkel

2.7.15

(kan bare spørres en vei)

første

_____

den første parameteren til en funksjon

den førstefødte ungen til arveprinsen av Nederland

førstegrads forbrenning

det er et teit navn som ingen vil huske hva står for om et halvt år

2.7.16

(kan bare spørres en vei)

prosess

_____

kjemisk reaksjon

kjernefysisk eksperiment

prosess i operativ systemet

det er et teit navn som ingen vil huske hva står for om et halvt år