

**Lab 3**

* Calendar

Name: Rasmus Muhrbeck

Date: 2020-11-18

**A lab report in the course DT555A Programming in C**

Table of Contents

[1. Introduction 1](#_Toc51231182)

[2. Design 2](#_Toc51231183)

[3. Implementation and Test 5](#_Toc51231184)

[4. Results and discussion 6](#_Toc51231185)

# 1. Introduction

This lab was made to practice further algorithm development. This is done by implementing the things we have learnt to harder and more complicated problems. This lab is not as trivial due to that the problems needs to be divided into more sub-problems.

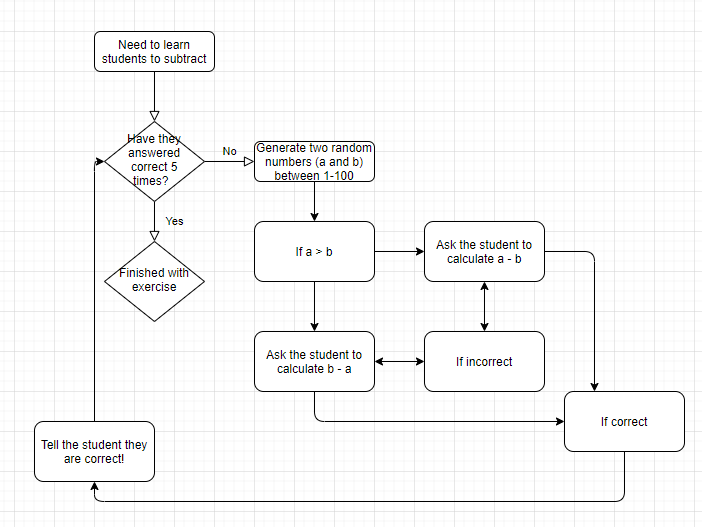
# 2. Design

**Task 1 Grade 5: Computer Assisted Instructions**

When first looking at the problem I concluded that the previous lab could be used to an extent to complete one of the sub-problems, the practice part. For it to work on the test-part of the solution it needed a small edit.

For the main menu I concluded that I only need to read the input of the user to know where he/she wants to proceed.

For the practice part thought was to have a counter that counts the amount of correct answers given by the student, and when it reached a certain number (5 in flowchart) it would be finished. While the student hasn’t answered enough correct guesses there would be 2 random numbers generated, lets call them a and b. To implement all three tests in one function I also included the users’ choice as a parameter. This meant that I could alter whether the user was given addition, subtraction or both. In the case of negative numbers and the fact that my primary school students don’t know negative numbers, I have to check if random number a is bigger than random number b. If that is the case, the program will ask the student to calculate a – b, and in the other case what b – a is. If the calculation is incorrect, the program will state that it was wrong and go back to asking the student to calculate the same a – b, or b – a, depending on the case. If the calculation is correct, the program will congratulate the student and after that check if the student has answered correct a certain amount of times. The program will then ask the student to calculate another two random numbers a and b. This will continue until the correct amount of answers required is achieved, which then will end the program.



For the testing part the same logic was applied with implementing the users’ choice an as input parameter. But instead of needing to answer correctly 10 times, the test consists of 10 questions and will end after 10 answers. I implemented the test so that it gives a return value of how many times the user answered correctly. This made it possible to calculate the percentage of the test result.

# 3. Implementation and Test

# 4. Results and discussion

The two pseudocodes that I described above have been tested and are working as intended. I like the idea of first writing a pseudo-code to simplify the program and break it down into smaller problems. This is a great way to get an overview of how one needs to structure the program to solve the task. By doing that the problems got quite easy to solve.