UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science
COS132 - Imperative Programming
Practical 3

1 Introduction

Deadline: 9th of April, 20:30

1.1 Objectives and Outcomes

In constructing programs, it is necessary to provide the means to create responsive and reactive programs that might respond to changing situations. In this respect, control structures like the if statement and its companion else are the first step towards creating responsive and dynamic programs that are not simply linear in nature. In addition, this practical will also introduce a number of concepts such as formatting output for users,library use and working with random numbers.

This practical will consist of 3 tasks and you will be required to complete all of them as part of this practical. You are also advised to consult the Practical 1 specification for information on aspects of extracting and creating archives as well as compilation if you need it. Also consult the provided material if you require additional clarity on any of the topics covered in this practical.

1.2 Structure of the Practical

Each task is self contained and all of the code you will require to complete it is separated. However you will only submit to a single slot: Practical 3. The practical will evaluate your tasks separately and give you feedback and your marks for each task. Each submission considers all three tasks so be sure to submit all the required code.

1.3 Submission

Submit your code to Fitchfork before the closing time. Students are **strongly advised** to submit well before the deadline as **no late submissions will be accepted**.

1.4 Plagiarism

The Department of Computer Science considers plagiarism as a serious offence. Disciplinary action will be taken against students who commit plagiarism. Plagiarism includes

copying someone else's work without consent, copying a friend's work (even with consent) and copying textual material from the Internet. Copying will not be tolerated in this course. For a formal definition of plagiarism, the student is referred to **http://www.ais.up.ac.za/plagiarism/index.htm** (from the main page of the University of Pretoria site, follow the *Library* quick link, and then click the *Plagiarism* link). If you have questions regarding this, please ask one of the lecturers, to avoid any misunderstanding.

2 Libraries

Up to this point, the only library you would have been using in practicals thus far is the iostream library. This library is necessary for all of the input and output operations like cout and cin in C++. However, there are many more libraries available that can add additional functionality to your programs without requiring you to code that functionality yourself.

One such library is the cmath libary (or math.h). It is designed to provide functions that perform many common mathematical tasks, like computing a square root of a number, raising numbers to exponents and trigonometric calculations.

Another useful library is the iomanip library which provides many useful formatting methods to produce well formatted program output.

In this practical, you will be making use of libraries; however, be aware that libraries are meant to supplement your coding and not to replace it.

The code provided in future will indicate which libraries are intended to be used to complete the practical or the specification will indicate it otherwise.

2.1 Mark Distribution

Task	**Mark**
Math Library	5
Output Formatting	5
Control Structures	10
Total	**20**

3 Practical Tasks

3.1 Task 1: Math Library and Expression

In this task you will be required to make use of the cmath library to complete a complex mathematical expression as well as the iomanip library for output formatting. Both libraries provide a lot of functionality with regards to easier and more well formatted output.

You are provided with a makefile and a file named task1.cpp. Open and study this

file to see it. The file is empty besides the skeleton that you are now used to except that the iomanip library has been declared already for you. Declare the math library for yourself.

Your task is to implement the mathematical formula presented below in C++ code, using the cmath library to resolve some of the calculations as well as do some numerical string formatting.

$$f(x) = \frac{cos(x^2)}{5 + 2 * cos(x)} + \frac{sin(x^2)}{5\pi} + \frac{tan(x)}{cos(x) + sin(x)} \tag{1}$$

Your program must first prompt the user for a value of x before then displaying the value of the calculation. You must format the output such that only five decimal values are shown. The 5th digit should be rounded up if the 6th digit, after the decimal point, is above 5. You are advised to use M_PI, the pi constant provided by the math library. Here, only 5 digits of the answer are displayed. The actual answer sum itself is unmodified.

An example is presented below:

```
Enter a value of x: 2.53
The answer is: 3.17035
```

3.2 Task 2: Output Formatting

In this task, you are going to be using the iomanip library to produce some output according to a strict criteria as well as get string input, via the string library, from the user.

You are provided with a makefile and a file named task2.cpp. Open and study this file to see it. The file is empty besides the skeleton that you are now used to. You will need to provide the correct definition for the libraries yourself.

You will first prompt the user to enter their name. You will then extract this name using getline() and store it in a string variable, called name. Note that there is a limit on how long a name can be in this instance. A name, when displayed, will display at most 7 characters but must always display 7 characters. If a name is fewer than 7 characters, you must provide a blank mask using the '*' character. You can assume names longer than 7 characters will not be used. Hint: You will need to make use of other functions from the iomanip library to create the mask.

Your program must prompt the user 3 times for inputs. Remember to end your outputs with new lines.

Examples of this is presented below (as separate executions of the same program).

```
Enter your name: Hanzo
Howdy: **Hanzo
Enter your name: Bastion
Howdy: Bastion
Please enter your name: Ana
Howdy: ****Ana
```

3.3 Task 3: Control Structures

In this task, you are going to be using the cstdlib (stdlib.h) to create random numbers and then use control structures to react to them.

You are provided with a makefile and a file named task3.cpp. Open and study this file to see it. The file is empty besides the skeleton that you are now used to. You will need to provide the correct definition for the library yourself .

You will first have to create an input prompt asking for the user to enter their student number. This will be saved as an integer value in a variable. If value of their student number is an even number, then display (on one line) a random 2 digit number from 10 to 99, followed by the message **is ready to take on COS 132!**. The random generator should be given the seed of 14.

Otherwise, then display (on one line) your student number followed by the message **is really excited for COS 132!**

Your program must ask for two prompts for input. Remember to add newlines to your output.

Examples of this is presented below:

```
Enter your student number: 12345678
23 is ready to take on COS 132!
Enter your student number: 17116513
17116513 is really excited for COS 132!
```

4 Submission Requirements

Your submission requirements for this task are:

- makefile
- task1.cpp
- task2.cpp
- task3.cpp

You will have a maximum of 10 uploads for this task.