UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science
COS132 - Imperative Programming
Practical 2

1 Introduction

Deadline: 2nd April, 20:30

1.1 Objectives and Outcomes

The objective of this practical is to introduce the basic elements that comprise programs. Variables, of various data types, arithmetic operations, mathematical expressions are all the basic elements from which all programs are constructed. Additionally, an introduction to pointers, an important language feature of C++ is presented as well.

This practical consists of 2 tasks and you will be required to complete it 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

The practical will consist of a number of tasks. Each task will require its own code based on the requirements of that task. Each task will be self contained so that code and data required for one task will not be required for others.

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.

1.5 Mark Distribution

Task	**Mark**
Data Types and Operations	15
Pointers	5
Total	**20**

2 Practical Tasks

2.1 Namespaces

In C++, namespaces are used to define the scope or context of a function. We use namespaces to allow our code to refer to functions contained within a namespace so that even if there are multiple functions with the same name, we can use their namespaces to differentiate them. In this practical, and in future practicals, we will assume the namespace std is considered which makes the task of input/output much easier.

You are provided code organised into separate folders. Please do not mix up which code is in which folder as this is task specific.

2.2 Task 1

In this task, you are provided with a file called t1.cpp and a makefile. You will notice that aside from the skeleton of the program, the file is otherwise empty except for some print statements at the end. Pay attention to these lines as they will indicate the types of the variables you will need. You will need to declare a number of variables according to the requirements below:

- A string variable containing the sentence "programming is fun"
- A char variable holding the character "X"
- Two integer variables holding the values of 75 and 29 respectively
- Two bool variable, the first set to true and the other set to false
- A constant double variable set to 1337.42

Once you have declared and initialised the variables, run your code along with the given cout statements to make sure that everything is working. Do not make changes to the cout statements. The output of this should be:

```
X comes after W in the alphabet
The sum of sixty and fifteen: 75
The multiplication of fourteen and a half and two: 29
Learning programming is fun
1 is the value of True
0 if the value of False
The constant has a value of 1337.42
```

After this, you will then need to complete the following requirements by adding the appropriate code after the first set of cout statements. You should declare all your variables together.

- Declare 4 integer variables named A,B,C,D respectively
- Set variable A to 14
- Set variable D to 31
- Set variable B to the product of variable A and variable D
- Set variable C to the division of variable D by variable A
- Add 1 to variable A
- Subtract 2 from variable D
- print out the value of variable A on a new line
- print out the value of variable B on a new line
- print out the value of variable C on a new line
- print out the value of variable D on a new line
- print out the sum of variable B and variable C on a new line

Once the code has been filled in according to the above specification

1. Compile the program using the makefile provided.
2. Run the program using the makefile. If you have provided the correct types, the program will execute and display several lines of text.

2.3 Task 2

2.3.1 Pointers

Pointers are an important part of the C++ language. They provide the programmer with a great deal of control over the use of memory, dynamic memory in particular, and enable programmers to precisely allocate (and deallocate) memory used by variables.

In task 2 you are provided with a file called t2.cpp and a makefile. The first requirement will be to declare a float variable, called myPi. You will notice some code under a comment

labelled, “User Input”. This will prompt the user to enter a value and store that input as the value of myPi. The later practicals will expand on this code. It uses the message: ”Enter a float number:”. You should not modify this code.

Now declare a float pointer variable, called piPointer and set its value to the reference of myPi. Then, output the value of piPointer with a new line at the end. After this, increase the value of myPi by 10.3 and then output the value of piPointer again with an end line. You should have two lines of output, both of which should be float numbers.

3 Submission

You are required to submit the following files in your submission archive:

1. t1.cpp
2. t2.cpp
3. makefile

One archive should contain the code for both tasks. You may submit either of the makefiles you were provided for the task.

You will have a maximum of 5 uploads for this practical. One submission accounts for both tasks.