Swinburne University of Technology Sarawak

COS10009 Introduction to Programming – Semester 1 / 2018

Standard Input / Output, Variable (Lab 01)

Core Task 1

To Do

"Hello World" Program:

The tutor will lead the following:

Run Code::Blocks / Quincy
Write, compile, run "*Hello World*" program
Save the program in the appropriate directory
Discuss the .C, .O, .EXE files in the directory where they were saved. Run the .EXE file from the file manager (Windows Explorer). It flashes onto the screen and disappears. Why? Fix it to wait for the user.
Next, prompt user for an input, write a function (e.g. *getch() / scanf()*) that ask and wait for input (char) before finishing and exiting.
Finally, write *printf()* statement to show that the character which the user types is actually received and will be printed on the screen.

Good programs have comments. Add the following comment block to the top of program:

```
/* FILE: Hello.c
 * Lab exercise 1
 * Author: <your name, student number>
 * Last Modified <today's date and time>
 */
```

Core Task 2

To Do

Debugging:

Download Syntax_errors.c and Logical_errors.c from BlackBoard.

Debug and fix all the errors in both programs. Try to familiarise yourself with some of the common syntax error messages shown, and how the IDE helps you to identify those syntax errors (eg: by double-clicking the error message will bring the cursor to the location where the syntax error occurs)

Note: To ensure a program is free from logical or run time errors, it is still necessary to test programs even after the compiling and linking stages have been successful?

*Hints : **%i** means print an integer (whole) number.*
__%f__ means print a floating point (real) number.
__%%__ means print a %

Core Task 3

To Do

Variables:

To explore this topic, we will create a program that will:
■ ask the user to enter their name, age, and the current year,
■ calculate the year the user was born, and
■ output a greeting message and the year user was born.

Procedure: Main

Declaration of variables:
- age (to store an Integer value)
- name (to store a String value)
- yearNow (to store an Integer value)
- yearBorn (to store an Integer value)

*You are encouraged to declare the variables with a unique name other than the suggested variable names given above.

Steps:

1: Read the user's name (a String, prompt with 'Please enter your name: ') and store it in the name variable.

2: Read the user's age (an Integer, prompt with 'How old are you this year? : ') and store it in the age variable.

3: Read the year (an Integer, prompt with 'What year is this year? : ') and store it in the yearNow variable.

4: Calculate the year the user was born (year - age), and store it in the yearBorn variable

5: Output the message, 'Hello ', name, ', you were born in ', then yearBorn

Note:

- *Four primitive data type in C are int (integer), float/double (floating-point), char (character) and Boolean (true/false)*
- Rules for selecting a valid identifier (variable/constant):
    - must begin with an alphabetic character or underscore
    - may contain only letters, digits and underscore (no special characters, no spaces)
    - C keywords are not allowed be used as identifiers
    - case sensitive

Vital Task

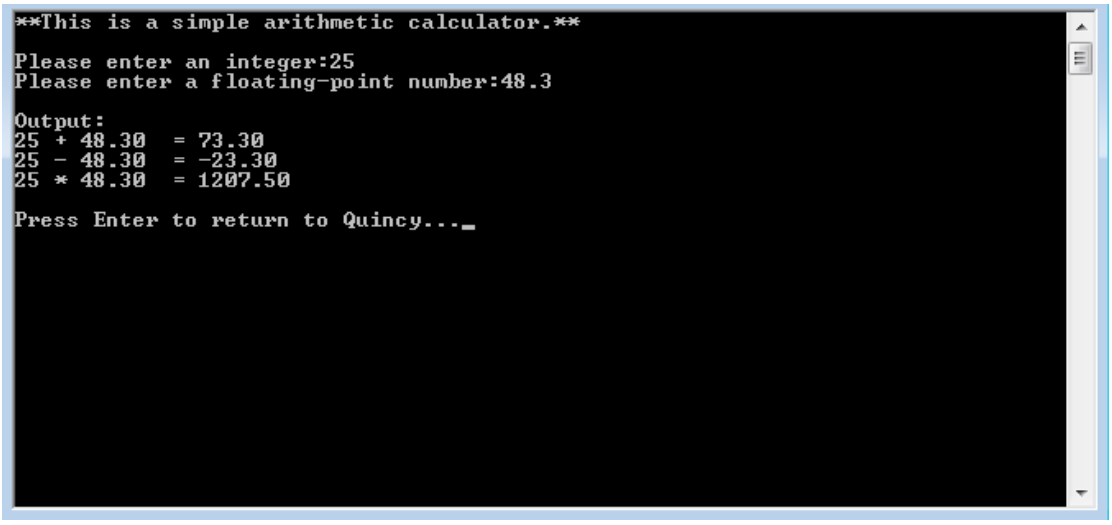
To Do

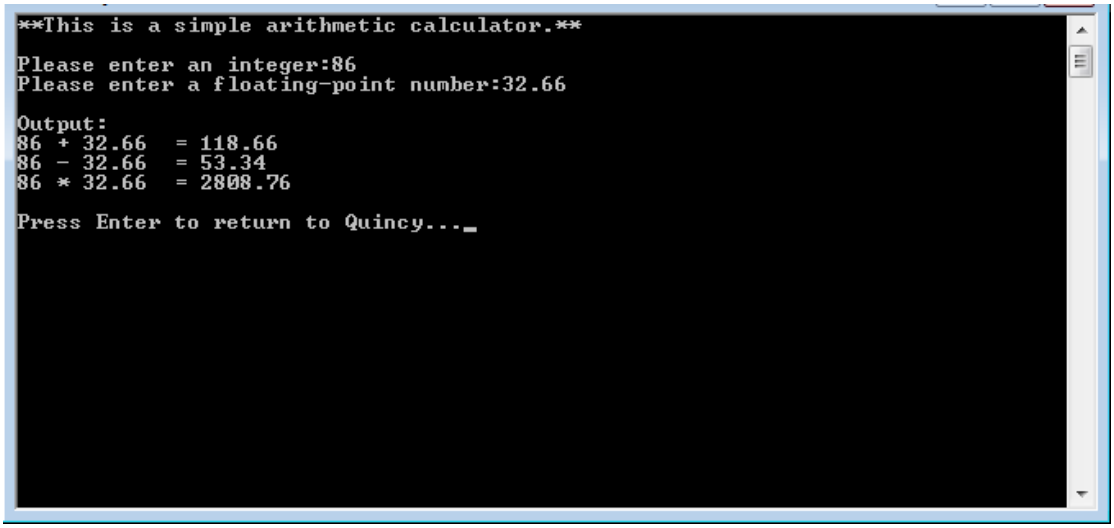
Handling User Input:

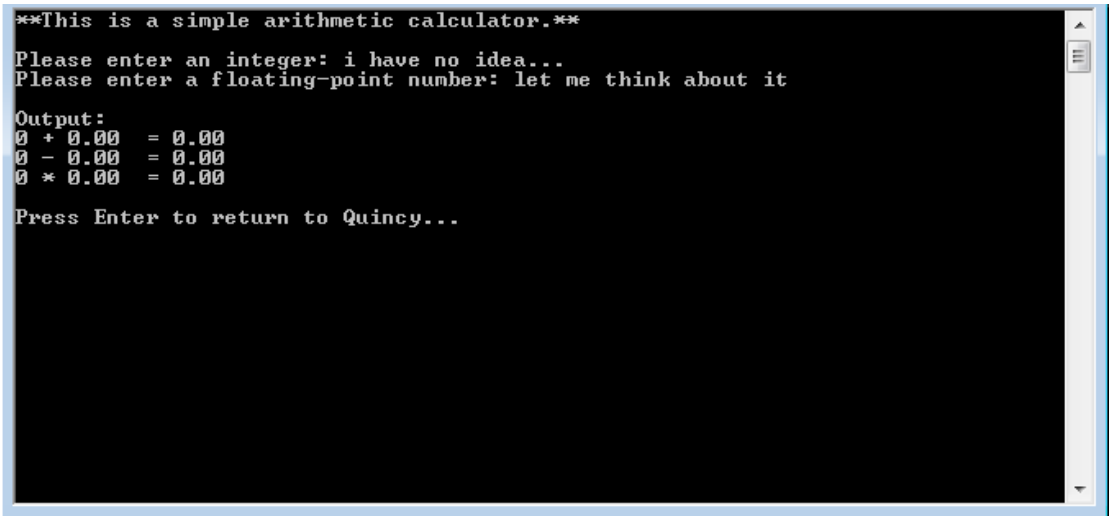
Reading values from the user is a common task, and one that will require some validation to ensure that the program works successfully. For example, if users are asked to enter an integer, the program will crash if they do not enter a valid number.

Your task is to develop a simple arithmetic program that reads an integer and a floating-point number and display the result of addition, subtraction and multiplication of those two numbers to the screen. You need to ensure that your program will not crash if the user enters an invalid input.

Examples of program execution:







Hints: In C programming, you may consider Alphabet to Integer function atoi(); available in stdlib.h

Note: There are multiple possible solutions to solve this task.

Challenge Task

To Do

Limitation and Precision:

Design and construct a C program to display the amount of memory allocated for various data types such as short integer, integer, double, long, char, boolean etc (Hints: use sizeof() function).
This program should also demonstrate the maximum limit of various integer data types (the largest and smallest number that can be assigned to an integer type variable) and the precision issue of float-point number. This task requires your understanding on how data types are represented by the computer and the size of computer memory to be allocated for each data type.

Exploratory Task

To Do

Design and write a C program that relates to the topics covered in this lab. The program should demonstrate your ability to use what you have learnt to explore associated concepts that go beyond the scope of this unit.