Workshop 1 – Exercises

1) Playing with escape sequences

In C programming language we can use some special characters with backslash ('\'), they are known as escape sequences. An escape sequence can perform special function in the output (printf) statement. Some popular escape sequences are:

Escape Sequence	Description
\a	Audible alert
\b	Backspace
\n	New line
\r	Carriage return
\t	Horizontal tab
****	Backslash

Use the search engine to find many other escape sequences (later). Lets start with a simple task now. Open DevC++ and create a new C program. Include the stdio.h header and add a main function. Within main(), type the following lines of codes. Predict the output before running your code and check if the output on screen appeared as you expected or not.

- `printf("Hello\nWorld\n");`
- `printf("Hello\\nWorld\n");`
- `printf("Hello\\\nWorld\n");`
- `printf("Hello\tWorld\n");`
- `printf("Hello\bWorld\n");`
- `printf("Hello\rWorld\n");`

Understand the use of escape sequences by viewing the output. Ask your tutor if you need explanation.

2) Simple calculations using arithmetic operators

Consider the following table which contains different operators that we can use for basic arithmetic operations involving two numbers:

Expression	What does it do
x + y	Adds y to x
x – y	Subtracts y from x
x * y	Multiplies x and y
x / y	Divides x by y

Now, write a program (remember to use the *.c file extension) so that:

a) The user can enter a floating-point number (e.g., 1.456), which represents the radius of a circle.
   Tip a: You need to select the type of the variable used to accommodate the value entered.

b) The program computes and prints onto the screen the circumference of the circle, followed by a line-break, and then the area size of the circle, followed by two line-breaks. All numeric outputs should be made to a precision of three digits after the "."
   Tip b1: The printf command uses different formatting symbols (e.g., %d for decimal numbers, or %s for strings) to embed variable contents of different types. Choose the correct formatting symbol you need to use with printf to output a floating point number of the required precision.
   Tip b2: The computation mentioned above requires the use of both variables and a constant (π).

3) Recognising and avoiding mistakes

Syntax errors are mistakes where commands and formal symbols were incorrectly written in the source code. Those who are new to programming tend to make syntax errors more often as the programming language and its structures are still unfamiliar. Attention to detail will get better as you write more and more programs and acquire a routine of coding correctly through practice. Syntax errors will usually lead to problems during compilation and prevent code from running at all. Look at the following program, which will not compile because it has a number of syntax errors. Use a pen to mark any syntax errors you can find.

```
#include <stido.h>
int main()     {
      float x,y; //initializing two variables
      Printf(“Enter 1st number: “);
      scanf(“%f”,x); //taking first input
      printf(“Enter 2nd number: “)
      scanf(“%d”,&y); //taking second input
      sum = x+y; //computing sum of two inputs
      prinft(“The sum of %f and %f is: /n %f,”x,y,sum);
      return 0;
```

4) Relational operators within a while loop (Challenge)

Write a program using a while loop that keeps asking the user repeatedly for an integer input, after which it evaluates this input to tell whether the number entered was a positive or a negative number and prints out the number and the result of the evaluation on screen (e.g., "-5 is a negative number."). If the user enters the answer to life, the universe, and everything[1], the program should end by breaking the loop and executing the return statement that should be sitting at the end of the main function.

Tip: For this task, you need to decide on the number and types of variables needed for your program. You should also use relational operators within the various control structures to evaluate the entered value and a) prepare the correct screen output and b) breaking the loop, when appropriate.

[1] If you don't know the answer to life, the universe, and everything, use a search engine of your choice to find out!