Workshop2 - Exercises

Consider the exercises below. Some involve writing a program to implement the specifications outlined in the task. Others have code examples for you to inspect and debug. As usual, tasks 1-3 are easier and have some guidance to help you along, whereas task 4 is a lot more challenging and requires you to do some research into the underlying problem and apply what you've learned so far more independently. Try to find solutions on your own as far as you can. If you get stuck, you can ask your workshop tutor for advice. Don't worry if you do not manage to complete all tasks within the workshop time. You can finish the sheet on your own time and sample solutions will be made available on Canvas before the next workshop.

1) Create a user menu using if-else

Once again, let's start with a simple task: Open DevC++ and create a new C program. Include the stdio.h header and add a main function. Use scanf to elicit a user input (integer 0, 1,2,3 or 4) and then use the if, else if and else command to process that input. Choose four different calculations to be performed at different key press of user. For example, if the user presses 1, it takes another input for radius and computes circumference of circle. If the user presses 2, it takes an input integer and computes its square root and similarly some other functions at 3 and 4.

As a result the program should show the menu and ask for user input when it's started and then takes input required for computation as wanted by user and finally displays result of calculation, and after this the menu will be shown again to repeat.

Note: If the user enters a 0, the program should quit (exit the loop).

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.

Tip: You will need to get further inputs from the user before you can actually compute something meaningful. Think carefully about the data types of the variables required for each result.

2) Create a user menu using switch

Repeat the above task using switch command. Use character input with switch (a,b,c,d and e). If the user enters e, the program should quit (exit the loop).
Make a little bit of change in the output as well. After the program outputs the result of the computation, it should wait for the user to press a key and then go back to the menu.

Tip: Use the command getchar() to pause execution and wait for a random user key press.

3) Recognising and avoiding mistakes – logic errors

a. Logic errors can occur where commands and formal symbols were written correctly in the source code but the code does not work as intended. That means the code will compile and execute but the results are wrong or not as intended (They are commonly called "bugs"). Logic errors are a lot harder to find and eradicate than syntax errors, and will often require the tracking of variable contents and control structures as the program executes. Therefore, using a debugger is advised. Look at the following program, which will compile but produce a wrong result. Its purpose is to take two integer numbers a and b, and then compute the following equation: result = $(a+b)^2 + 4ab$. It comes up with 360, which is wrong. Load the program into DevC++ and run with the debugger active to track. down and eliminate any mistakes made. What is the correct solution, given a=5 and b=18

```
#include <stdio.h>
 int main()
 {
// declare and initialise variables and constants
const int a = 5;
const int b = 18;
int result = 0;
// compute (a+b)^2
 result = a + b * a + b;
 // add 4ab and update result
result = 4 * a * b;
printf("(%c + %d)^2 + 4*%d*%d is %d", a,b,a,b,result);
getchar();
return 0;
}
```

b. What will be the output of the following computations if performed by C?

i. 34 + 12 / 4 – 45

ii. 12 + 3 – 4/2 < 3 + 1

Tip: Search for operator precedence in C

4) Decimal to binary conversion (Challenge)

Write a C program which takes a positive integer input from the user and outputs its binary equivalent after conversion. For example, if a user enters 5, it displays its binary equivalent, i.e. 101.

This task will require you to do some independent research into number bases, division with remainder, and loops (which we will cover properly next week).

Tip: Use arrays, a loop, conditional statements and the modulo operator(%), which yields the remainder of an integer division.