17

C Programming Worksheet

Activity: Work through the following exercises and use the time available to recall and practice C programming concepts and implementations.

Purpose of this Activity

This activity will repeat and practice common C programming concepts implementations in preparation for the programming test. You will need a C compiler to work through these tasks.

Note however that a Raspberry Pi or Gertboard are not required. For a suitable C compiler and integrated development environment (IDE), visit SOL for a link for the CodeBlocks application.

Learning Outcomes

- To be able to translate specification requirements into a program design and implement and test this as an application.
- To be able to apply programming concepts in C.

TASK 1:

Recall the key elements in C for the following programming concepts:

- Program structure / sequence
- Iteration / Loops
- Conditional Statements and Branching
- Data types and Variables and Arrays
- Mathematical Operators, including comparison
- Boolean Combinational logic
- Including header files of external libraries

TASK 2: **#include, main(), printf()**

Write, compile and execute the (smallest) C program to display “Hello World!” on a console window.

TASK 3: **for-loop, integers, double**

3.A) Using a ‘for’ loop, write C code to display the integer numbers between -10 and +10.

3.B) Using a ‘for’ loop, write C code to display the double precision numbers between 10.0 and +10.0 in increments of 0.5 .

3.C) Write the code to display the inverse, i.e. from +10.0 to -10.0 in 0.5 decrements.

TASK 4: while-loop

4.A-C) Implement tasks 3.A-C with a while loop instead of a 'for' loop.

Remember the 'for'-loop syntax:

```
for ( <set startvalue>, <while-condition>, <increment> )
{
        <statements …>
}
```

And for the 'while'-loop:

```
<set startvalue>
while( <while-condition> )
{
        <statements …>
        <increment>
}
```

TASK 5: variables

What does the following code do? – Analyse it first, then set your expectations. Then implement, compile, build and run it to see if it matches your expectations.

```
int main( void )
{
        float startValue = 5.0;
        float stopValue = 10.0;
        for (i=startValue; i<=stopValue; i=i+0.5)
        {
                printf(" %g^2 = %g \n", i, i*i);
        }
}
```

TASK 6: arrays

6.A) Write a for-loop that displays a range of 12 integer values stored in an array (of type integer).

6.B) Write a for-loop that display a range of 10 double precision values stored in an array (of type double).

Reminder:

To declare a 3-element array of type integer (which is empty after declaration):

```
int myArrayName[3];
```

To declare and fill a 3-element array of type integer:

```
int myArrayName[3] = {1,2,3};
```

TASK 7: Gertboard interface

Write out the C variables and functions to interface to the Gertboard that we have used in the last few weeks. These are defined in the gertboard library. You may use the Gertboard instruction manual to find out more (this is installed on the Raspberry Pi and also available online). Add a brief description to remind you what they do.

You will be able to use this list as reference in your final open-book programming test.

For example:

#include “gb_common.h” – Includes the header file for the Gertboard library functions so that the compiler knows what variables, constants and functions are available from the library when compiling the code to an object file.