C2

## Functions and Control Flow

In this exercise you will build a function plotter in three stages. It will reinforce what you have learned in the first lab, introduce you to calculations using the math library, to writing your own functions with local variables, and to using conditions and loops to control the flow of your code.

```
x = 0000019132, y = 0.92 |
x = 0000019133, y = 0.95 |
x = 0000019134, y = 0.98 |
x = 0000019135, y = 0.99 |
x = 0000019136, y = 1.00 |
x = 0000019137, y = 1.00 |
x = 0000019138, y = 0.98 |
x = 0000019139, y = 0.96 |
x = 0000019140, y = 0.93 |---
x = 0000019141, y = 0.89 |
x = 0000019142, y = 0.85 |
x = 0000019143, y = 0.79 |
x = 0000019144, y = 0.73 |
x = 0000019145, y = 0.67 |
x = 0000019146, y = 0.60 |
x = 0000019147, y = 0.53 |
x = 0000019148, y = 0.46 |
x = 0000019149, y = 0.39 |
x = 0000019150, y = 0.33 |---
x = 0000019151, y = 0.26 |
x = 0000019152, y = 0.20 |
x = 0000019153, y = 0.15 |
x = 0000019154, y = 0.10 |
x = 0000019155, y = 0.07 |
x = 0000019156, y = 0.04 |
x = 0000019157, y = 0.01 |
x = 0000019158, y = 0.00 |
x = 0000019159, y = 0.00 |
x = 0000019160, y = 0.01 |---*
x = 0000019161, y = 0.03 |
x = 0000019162, y = 0.05 |
x = 0000019163, y = 0.09 |
x = 0000019164, y = 0.13 |
x = 0000019165, y = 0.18 |
x = 0000019166, y = 0.24 |
x = 0000019167, y = 0.30 |
x = 0000019168, y = 0.37 |
x = 0000019169, y = 0.43 |
x = 0000019170, y = 0.50 |---
x = 0000019171, y = 0.57 |
x = 0000019172, y = 0.64 |
x = 0000019173, y = 0.71 |
x = 0000019174, y = 0.77 |
x = 0000019175, y = 0.82 |
x = 0000019176, y = 0.87 |
x = 0000019177, y = 0.92 |
x = 0000019178, y = 0.95 |
x = 0000019179, y = 0.98 |
x = 0000019180, y = 0.99 |---
x = 0000019181, y = 1.00 |
```
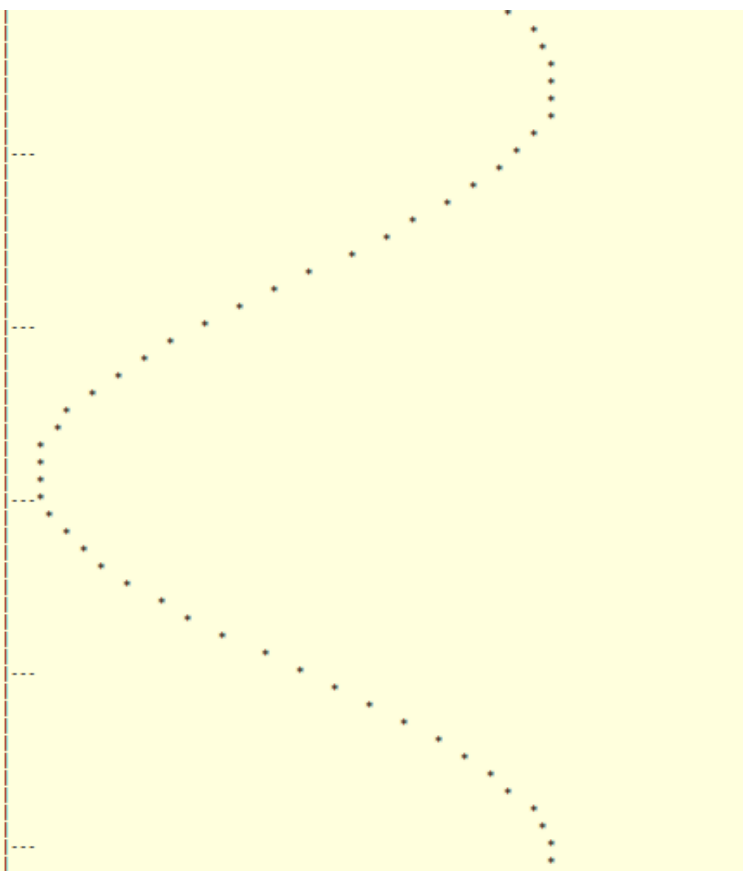
Schedule

Preparation Time	:	3 hour
Lab Time	:	3 hours

Items provided

Tools	:	
Components	:	
Equipment	:	
Software	:	`gcc`, `notepad++` or equivalent text editor

Items to bring

Course Textbook - *Identity card*

Laboratory logbook

Before entering the laboratory you should read through this document and complete the preparatory tasks detailed in section 2.

> **Academic Integrity** – *If you wish you may undertake the preparation jointly with other students. If you do so it is important that you acknowledge this fact in your logbook. Similarly, you will probably want to use sources from the internet to help answer some of the questions. Again, record any sources in your logbook.*

You will undertake the exercise working with your laboratory partner. During the exercise you should use your logbook to record your observations, which you can refer to in the future – perhaps to write a formal report on the exercise, or to remind you about the procedures. As such it should be legible, and observations should be clearly referenced to the appropriate part of the exercise. As a guide the ✎ symbol has been used to indicate a mandatory entry in your logbook. However, you should always record additional observations whenever something unexpected occurs, or when you discover something of interest.

For each Task you should create a new directory so that you have a working version of every program at the end of the lab. Remember to place comments in your code.

You will be marked using the standard laboratory marking scheme; at the beginning of the exercise one of the laboratory demonstrators will mark your preparatory work and at the end of the exercise you will be marked on your progress, understanding and logbook.

Notation

This document uses the following conventions:

✎	An entry should be made in your logbook
⚠	Care should be exercised
`command input`	Command to be entered at the command line

1 Introduction

This lab introduces you to programming in C.

1.1 Outcomes

At the end of the exercise you should be able to:

- ▶ Write a C program from scratch.

- ▶ Write your own functions.

- ▶ Use conditions and loops in the design of your programs.

- ▶ Perform calculations with formatted output.

2 Preparation

Read the laboratory work section below, read up on any new material (in the Textbook and/or on-line) you require to accomplish the tasks and make notes in your logbook including the source from where you got the information, so that you can quickly go there again if something does not work as expected. For this lab you will not receive skeleton code, so you should be prepared to open an editor and write a C-program from scratch, based on what you learned in the first lab.

At least you should have the following in your logbook before the start of the lab:

- ▶ Notes about declaring an unsigned long variable, how to initialize it correctly, and what format specifier is required to print such a data type.

- ▶ Notes on how to print a number with a fixed character width.

- ▶ Notes on how to create an endless loop in C.

- ▶ A formula that gives you the sine of a value scaled to the range of 0 to 1. (You should also have an idea how to change the frequency in your formula.)

- ▶ A draft of the `plotval()` function needed in Part 2. Consider to use a loop to print an appropriate number of spaces into the same line.

3 Laboratory Work

3.1 Part 1

Write a C-program that in its `main()` function declares a local unsigned long variable named `x` and initializes the variable with a value of zero. After initializing `x` the program should enter an endless

loop. Within the loop the program should increment x (i.e., add one to x) and print the value of x with a fixed width.

Compile and start your program on the command line. You can stop your program[1] by typing Ctrl-C in the terminal.

Debug your program and make sure it easy to read for humans (correct indentation, comments), then print it out add it to your logbook.

Make a copy of the code you have developed so far. Include the math library (`math.h`). Note, that you will need to let the compiler know that it should link with the math library, by using the option `-lm` before the output option.

Declare in `main()` a variable named y of type float. In your endless loop, modify the printing of x such that it does *not* end the line. Following the printing of x, calculate the value of y as the sine function of x multiplied by $\frac{\pi}{180}$. Define `PI` with the preprocessor. After the calculation of y, print it with fixed width and two or three digits after the point. Let this end the line.

When this works, modify the formula such that the value of y is shifted and scaled to fall into the range 0...1.

Verify that the scaling works, then define a constant FREQ that can be used in your formula to increase the frequency of the sine function (assuming x is a time), you can set it to 1 and adjust it later.

Debug your program and make sure it is easy to read for humans, then print it out add it to your logbook.

3.2   Part 2

In this section you will visualize the function calculated in the previous part. Make a copy of your code from Part 1.

Declare and define a function `plotval()` that takes two arguments and returns nothing. The first argument is a floating point value in the range between 0 and 1, and the second argument is an integer, the width of the terminal that is available for plotting. The latter is given in characters, and will depend on the width of the terminal available. Define a constant for the width of the plot and use it in calls to the `plotval()` function as second argument; a good value may between 50 and 60, depending on how you have formatted your numbers in Parts 1.

The function `plotval()` should print a symbol (e.g. '*') at the current position in the line if the first argument is 0, plot it shifted to the right by the number of characters given in the second argument, if the first argument is 1, and position the symbol between these extremes proportionally to the value of the first argument, if it is between 0 and 1. This function should also end the line.

Debug your program and make sure it is easy to read for humans, then print it out add it to your logbook.

---

[1] As well as most other programs that have been started on the command line.

3.3 Part 3

Make a copy of your code from Part 2.

Modify your main loop such that it shows a tick-mark following the numbers at regular intervals (The cover page shows a tick mark at every ten values). Use an `if...else` condition togehter with the modulo-operator '`%`' to print either the mark or an equivalent number of spaces. Define a constant that can be used to change the spacing of the tick-marks.

Debug your program and make sure it is easy to read for humans, then print it out add it to your logbook.

Improve the output formatting to your liking, print an illustrative piece of the curve and add it to your logbook.

Consider what determines the speed with which your program executes; make a note about your thoughts.

4 Optional Additional Work

Take a look at the terminal color control sequences, e.g. in this Stack Overflow Answer:

https://stackoverflow.com/a/23657072

or from here:

http://wiki.bash-hackers.org/scripting/terminalcodes

Improve the aesthetics of your plotter by using colour on the terminal. Plot more interesting periodic functions, e.g. summations of sine functions with different phase and frequency. You can add noise with the `rand()` function from the math library. Print out a representative part of an interesting curve together with its formula and add it to your logbook. Make sure you cite all sources of information you have used in the comments of your code and in your logbook.