Week 12, Lab 12	Weight: 1%	Due: End of your stream's week 12 lab session (via sync)

Pre-lab Preparation:

- **Week 1 - 11 Lectures, Week 1 - 11 Labs**

Lab Activities:

Remember to **sync** to obtain the lab starting code.

Exercise 1: How to prepare for the Final Test

Navigate to the directory: **~/p1.2015s1/lab12/working_copy/ex01/**

Complete the electronic Week 1 to Week 11 Knowledge Checklist, open **checklist.txt** in the **editor**. Once you have completed the checklist. Run a **sync**.

The lab TA will also provide you with a printed hand out of "Programming 1 Test and Exam and Preparation Guide". Review this document.

You will want to prepare your own unique materials for the open book final exam. Ensure you set aside time before the exam to prepare your notes, and complete any further study you require in preparation.

Exercise 2: Your hardest thing

Navigate to the directory: **~/p1.2015s1/lab12/working_copy/ex02/**

Identify one thing from your checklist that is the part of C programming that you are most unsure about. In the file **thing.txt** record what it is, work out where it is covered in the lecture slides, and at least one question you might ask your TA about the topic to help you understand it better. For example:

```
Thing: arrays
When was it covered in lectures:
What can I ask?
- How can I tell how long an array is?
- How can I return an array from a function?
Answer:
...
```

When you have completed **thing.txt**, go back and read the lecture slides about the topic. If afterwards you know the answer to your questions, update **thing.txt**

Exercise 3: Test your hardest thing

Navigate to the directory: **`~/p1.2015s1/lab12/working_copy/ex03/`**

Write a program that demonstrates your new understanding of the thing that you were uncertain of in Exercise 2. For example, if you were uncertain about how to return an array from a function, write a function that does so, and print its contents.

You will want to prepare your own unique materials for the open book final exam. Ensure you set aside time before the exam to prepare your notes, and complete any further study you require in preparation.

Once complete, have a TA review your completed exercises 1, 2 and 3 for this lab session. See the end of this document for the review questions.

The following exercises are intended to be challenging questions that will give you practice using a range of C features. You may want to complete some or all of them before the test. They will not be marked.

Exercise 4: Number converter

Write a program that can convert a number that the user enters into another base. You should support binary, octal, decimal and hexadecimal.

The program should take two arguments given at the command line. The first should be the number to be converted. The program should recognise the format of the number from its prefix.

The number is in binary if it starts **`0b`**, e.g. **`0b11111111`**.

The number is in octal if it starts with a leading **`0`**, e.g **`047`**

The number is in hexadecimal if it starts with **`0x`**, e.g **`0xff`**

Otherwise the number is assumed to be decimal.

The second argument should be a single character identifying the required output format: **b** for binary, **o** for octal, **x** for hexadecimal and **d** for decimal.

The program should print the converted number in the required format *not* including any prefix and return with success.

If incorrect arguments are given to the program it should print an error message on **`stderr`** and return with error.

You should not use any library function such as **`sscanf()`** or **`atoi()`** that is capable of converting a multi-character string to an integer in your implementation. Instead, convert the number by performing character arithmetic on the individual input characters.

Exercise 5: File analyser

Write a program that takes a filename argument on the command line and reports how many bytes are in the file, and whether the file is a text file or binary.

Extend your program to include a count of the number of 1 bits and 0 bits in the file.

Exercise 6: Game of life

Read the rules for Conway's game of life and implement it using the image world code provided. Try starting with random placement of 'live' pixels.

`http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life`

Exercise 7: Rabbits and lettuces

Another interesting 'life' game is rabbits and lettuces. Rabbits randomly move to adjacent squares. If the square is occupied by a lettuce they eat it. If they fail to eat a lettuce within *n* moves they starve. If a rabbit is adjacent to another rabbit they reproduce, adding a new rabbit to a free adjacent square. Lettuces grow at random. Visualise the evolving scenario. Graph the population density of rabbits and lettuces for different starting populations and different parameters such as lettuce growth and rabbit starvation rate.

Exercise 8: Idle RPG

Idle games play themselves without player involvement (but nevertheless produce 'amusing' output). Venerable examples are the original IdleRPG (an IRC bot) and Progress Quest (Google it, but don't get stuck playing Cookie Clicker all day).

Implement your own idle RPG. Use C structs to implement characters, monsters and items. Make sure interesting and amusing output is produced, indicating character progress and adventures.

You may find the Posix function **`sleep()`** helpful for implementing short delays.

Use file IO and Unix signals to allow a game to be suspended in response to **`SIGINT`** and resumed later.

Week 12, Lab 12 Submission:

Run the **`sync`** command to submit your completed lab work.

Shutdown your Raspberry PI by pressing **`ALT-CTRL-DEL`**. Power-down and pack up your Raspberry Pi kit.

Marking Criteria:

Have you completed each of the following? Have you submitted your code from lab?

Marking Criteria:	Week 12 Lab 12 Weight 1%	Yes	No
Ex 1:	Checklist complete?		
Ex 2:	**`thing.txt`** completed?		
	Confusion resolved?		
Ex 3:	C problem clarified though experimentation?		
Ex 4 - 8:	Completed to your own satisfaction?		
	Any programming difficulties investigated and resolved in the manner of Ex. 2 and Ex. 3?		

Next activity: Final Exam!