COMP2100 Workshop Week 4

1. Sketching some simple code with pointers
    a. Sketch the variables of the following piece of C code (two integers and two pointers).

```
int j, k;

int *p, *q;
```

    b. Where do the pointers point in the above C code? What values are in the integers j and k?

    c. Sketch the effect of the following C statements, assuming the variable declarations as above.

```
p = 0;

q = &j;

j = 15;

k = 23;

*q = 11;
```

    d. What would happen if you tried to execute the following statement, after all the above?

```
*p = 18;
```

2. Briefly explain each of the following pointer errors and its likely impact on your program.
    a. Uninitialised pointer.

    b. Dereference null pointer.

    c. Increment past end of array.

    d. Pointer to local variable that has gone out of scope.

3. Command line arguments and stdio. In this question, you will develop parts of a program that reads a text file and converts the alphabetic characters to lower case or to upper case depending on an option. Here is the command line syntax:

```
$ ./program option filename
```

    where option is either `-u` or `-l` and filename is the name of a text file.
    a. Write a snippet of C code to open the named file for reading using `fopen`. Don't forget to respond to errors.

b. Write a snippet of C code to check whether the option is `-u` or `-l` or invalid. If the option is invalid, your code should print an error message and call a `usage()` routine to remind the user of the command syntax. If the option is valid, set a boolean variable to tell you whether the option is –u (upper case) or –l (lower case).

c. Write a loop that reads the file character by character (using fgetc or getc), until EOF is encountered. As each character is read, check whether it is alphabetic and convert it according to the Boolean option, then print out the (converted) character to stdout. Hint: Look at man pages for tolower and related functions.

d. Write a code snippet to close the input file.

e. In your own time, complete your program by assembling the code snippets and writing the missing parts such as the usage() routine. Test the program.