Exercise 1 – Implementing a Stack

Due by the end of your Week-2 laboratory class. **(2 marks)**

This exercise is to be done during your week 2 laboratory class. When you complete the exercise show your work to your lab tutor to have your work marked and submit it according to the submit instructions below. The marking is based mainly on correct implementation and code readability.

You should implement your code in one file (e.g. main.c or main.cpp). Make sure your program has a header comment block containing the name of the exercise, your name and your student login (e.g. jfk01). If you prefer to implement your code in java, (rather than C or C++), please first discuss this with your lab tutor.

For this exercise you are to write a program (in C or C++ preferably) that reads a text file containing a number of words and displays the words on the screen in reverse order using a stack. A pseudo-code outline for the program is given below:

```
Begin main
      display a prompt for the file name
      read in the file name
      try to open the file
      if the file fails to open
           print an error message on the screen and exit
      fi
      do
           read in a word from the file
           if the file read fails
                terminate (break) the loop
           fi
           Push the word onto the stack
      od
      close the file
      while the stack is not empty
           display the top stack word on the screen followed by a space
           pop the top value from the stack
      elihw
End main
```

Do not implement the stack using a class or struct or with STL. The stack must be implemented using a fixed size array of words and an index integer for indicating the top of the stack. The stack array and index should be global variables. A word can be a string or a c-string (i.e. a character array). You can assume no word is more than 20 characters long. The stack functions (i.e. push(), top(), pop(), isEmpty()) should be implemented below the main() and prototyped above the main().

When you are finished, test your program using the provided text file named "Ex1.txt" and show your code and the output to your lab tutor to receive your mark. Also, submit your file via unix (banshee) using the submit command below.

```
$ submit –u login –c CSCI203 –a ex1 filename
```

where '*login*' is your UNIX login ID and '*filename*' is the name of your file.

If you are unable to attend your lab class and demonstrate your work on time due to circumstances beyond your control (e.g. sickness), contact the coordinator (Ian or Koren) to request an extension.