1 A program stores integers in a stack. The stack is represented as a 1D array of 30 elements with the identifier `Stack`

The global pointer `TopOfStack` stores the index of the last element inserted into the stack. `TopOfStack` is initialised to `-1`

(a) Write program code to declare `Stack`, initialise each element in the array with a null value and declare and initialise `TopOfStack`

Save your program as **Question1_N25**.

Copy and paste the program code into part **1(a)** in the evidence document.

[2]

(b) The function `Push()` takes an integer parameter. If the stack is full, the function returns `FALSE`. If the stack is not full, the parameter is inserted into the stack, the pointer is updated and the function returns `TRUE`

Write the program code for `Push()`

Save your program.

Copy and paste the program code into part **1(b)** in the evidence document.

[4]

(c) The function `Pop()` returns the next integer in the stack and updates the pointer as appropriate. If there is no data in the stack, the function returns the value `-999`

Write program code for `Pop()`

Save your program.

Copy and paste the program code into part **1(c)** in the evidence document.

[4]

(d) The main program generates 40 random integers between 0 and 1000 (inclusive) and attempts to insert each one into the stack using the appropriate function. If the return value from the function call indicates the stack is full, no more integers are generated and `"Stack full"` is output.

Write program code for the main program.

Save your program.

Copy and paste the program code into part **1(d)** in the evidence document.

[4]

(e) The procedure `FindValues()`:

- pops each integer from the stack until the stack is empty
- finds and outputs the largest number that was in the stack in an appropriate message
- finds and outputs the smallest number that was in the stack in an appropriate message.

Write program code for `FindValues()`

Save your program.

Copy and paste the program code into part **1(e)** in the evidence document.

[4]

(f) **(i)** Extend the main program to call `FindValues()`

Save your program.

Copy and paste the program code into part **1(f)(i)** in the evidence document.

[1]

(ii) Test your program.

Take a screenshot of the output(s).

Save your program.

Copy and paste the screenshot(s) into part **1(f)(ii)** in the evidence document.

[1]