Lab 3.2, Part A – Create a Stack for Doubles

Learning Goals

1) Develop your ability to program in C and use a Unix shell.
2) Develop your understanding of the stack data structure.
3) Develop your understanding of header files.
4) Develop your understanding of "blackbox" testing.

Your Task

Create **stack.c**, implementing the declarations in **stack.h**. (attached)

Then, test your implementation using **stack_test.c**. (attached)

Stack_test.c performs what is termed as a **black box** test. It does not see anything inside your stack.c implementation. However, stack_test knows what stack.c is supposed to do according to our interface rules. Stack_test rigorously tests many possible calls to stack.c, including popping an empty stack and overfilling a full stack. Good developers make black box testing a habit.

Debug until your stack.c passes all 9 tests. In part b of this assignment, you will use your stack to implement an RPN calculator. So, you will want a bug free stack!

Two implementation notes:
1) As with the parentheses assignment, set your max stack size to 100.
2) Import math.h in order to be able to use the constant NAN.

Sample Run, passing all tests.

```
$ gcc -o test stack.c stack_test.c

$ ./test
Passed 1
Passed 2
Passed 3
Passed 4
Passed 5
Passed 6
Passed 6
Passed 7
Passed 8
Passed 9
```