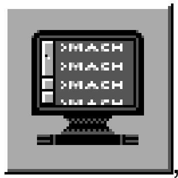
OOP Class Lab
RPNCalculator

Purpose: This lab is an exercise in basic Objective-C, creating a subclass, and writing methods. You will design a new object called RPNCalculator. This object uses the Stack object and implements a simple two-function calculator. You will write the methods for the new class, and you will write the main program to test the new object.

1. Start up a Terminal shell , and change to the **OOPClass**/**Labs**/**RPNCalculator** folder (type "**cd OOPClass**/**Labs**/**RPNCalculator**". Type "**pwd**" to check that you are there.).

2. Run the RPNCalculator program in the Solution directory (type "**Solution**/**RPNCalculator**") to see how it works.

3. For the new RPNCalculator class, the .h file and a skeleton .m file are provided. Use Edit  to add the necessary methods to the .m file. The required methods are:

- init (initializes the RPNCalculator instance, creating a stack instance for it to use)
- enter: (pushes the value passed as its argument onto the calculator's stack)
- add (adds top two values on stack, pushing sum onto stack)
- add: (calls add to add argument passed to value on top of stack)
- subtract (subtracts top two values on stack, pushing difference onto stack)
- subtract: (calls subtract to subtract argument passed from value on top of stack)
- printStack (prints values on the stack)

- allClear (clears all values on calculator's stack)
- free (frees the stack and the calculator itself)

4. Look at the skeleton main program RPNCalculatorMain.m provided. The comments indicate where you should supply code to send messages to your RPNCalculator instance to perform the various functions indicated. Again, using Edit, add the necessary code to RPNCalculatorMain.m to send these messages.

5. The Makefile and Stack files are already provided. Make the program by typing "**make**". Then run your program by typing "**RPNCalculator**".

6. If you have problems, you can sneak a peak in the Solution directory.

Extensions: Now add Multiply and Divide capability to RPNCalculator. Then maybe add a second Memory register, along with Memory Add, Memory Subtract, and Memory Recall functions.