91.102 : Exercise 1

Read

- Section 3.1
- Section 3.2
- Section 5.1

in Esakov and Weiss and do the following tasks.

1. Make a new directory, called ex1, for this exercise. Always make a new directory for every exercise as you may need work that you have previously done later in the course.

2. Download the starter Makefile from the Exercises page. This is a somewhat advanced Makefile and you will not understand it all. Do not let this worry you at all at this time.

3. Type "make" at the prompt. If you get an error then something has gone wrong with your download and you should make sure to fix it before going on.

4. Open a new file called stack.h in you favorite text editor. Add identifying comments to the file. Then add the preprocessor directives

   ```
   #ifndef <SOME VAR>
   #define <SOME VAR>

   #endif
   ```

   where <SOME VAR> stands for some well chosen variable name. I chose the following randomly generated name:
   H0f2a6614_6102_11e2_be81_005056963a9a.

5. Between the #define and the #endif line add the following type definition.

   ```
   typedef struct stack stack;
   ```

6. Between the #define and the #endif line add the following procedure declarations. Make sure they follow the type definition.

```
extern status init_stack(stack * const p_S);
extern bool empty_stack(stack const S);
extern status push(stack * const p_S,
                   generic_ptr const data);
extern status pop(stack * const p_S,
                  generic_ptr * const p_data);

extern status destroy_stack(stack * const p_S,
                            void (*p_func_f)(generic_ptr));
```

7. Type "make headers" at the prompt. Most likely you will encounter some oddly inscrutable errors. The problem is that you are making use of several types that are not defined in the C language and that you have not defined yourself. These types are status, bool, and generic_ptr. Open a new file called globals.h and put the triple of preprocessor directives in the file just as you did for stack.h. Make sure to use a different variable, though.

8. Between the preprocessor directives in globals.h add the following type definitions and save the file.

```
typedef enum {OK, ERROR} status;
typedef enum {FALSE, TRUE} bool;
typedef void* generic_ptr;
```

9. Type "make headers" at the prompt again. You should get the same inscrutable errors. To fix the problem, include globals.h in stack.h. That is, put the line

```
#include "globals.h"
```

   above all other lines in stack.h but between the original preprocessor directives. Type "make headers" again. If you experience a problem, stop and fix the errors.

10. Make a new file called stack_utils.h. Add the three pre-processor directives you always need to add. Then add the following code.

```
#include "stack.h"

extern status top(stack * const p_S,
                  generic_ptr * const p_data);
```

    Type "make headers" and fix any errors that you encounter.

11. Hand in your code using Bottlenose. Make sure to review the instructions for using Bottlenose on the course webpage.

12. If you had any problems with the assignment write out an explanation of the problems you encountered and hand the explanation in at the start of the next class period following the due date of the assignment.