PHYS 352 – Assignment 1

Due: Fri., Jan. 14

Submit code solutions for each of the problems below. Your C source files should be named "assignment1_X.c", where "X" corresponds to the question numbers. Include your name enclosed in C comment tags (ie: `/*YourName*/`) at the top of each program. Create a zip archive containing all of your code, name it "assignment1_YourLastName.zip" (with the appropriate name replacement) and copy it to your `/projects/e20271/student/[netID]/homework` directory by midnight on Friday. Work on this assignment individually.

1. **n! (1 pt.)**

   Write a program to calculate the factorial of a integer number whose value is specified via terminal input.

2. **Overflow Protection! (2 pt.)**

   In class we discussed the "getLotsOfChar" program, which reads characters from the terminal, stores them in an array and then outputs them back to the terminal. As written, the program's behavior is ill-defined if fewer than `ARRAY_SIZE` characters are input. Moreover, the program contains a weakness in that it does not check whether the data input by the user will actually fit in the allocated array. These issues can be addressed using the `fgets` function:

   ```
    char * fgets( char [], int size, FILE * )
   ```

   `fgets` stores at most size-1 characters from the input into the array argument and adds a terminating `\0` as the last element of the array. `fgets` returns `NULL` when `EOF` is read. Input characters beyond the specified array length are discarded.

   Rewrite getLotsOfChar3.c (from the examples) to use `fgets` in the conditional expression of the `while` loop. The call to `fgets` will look similar to :

   ```
   fgets(carray, ARRAY_SIZE, stdin)
   ```

   In addition, replace the cumbersome `putchar` loop with a simple `printf` statement. Ultimately, your code should not exceed 20 lines. Your program output should resemble :

   ```
   Please input some characters ...

   abcd
   You input: abcd

   efghijk
   You input: efghijk

   1234567890
   You input: 1234567890
   ```

```
fini!
```

3. **Stack Smashing!   (3 pt.)**

   This problem highlights the danger of buffer overflows. Declare two statically initialized character arrays, `a` and `b` , in that order. Set `a` to the string “hello” and `b` to “there”. Print `a` and `b`. Then create a `for` loop that runs over positions 6-17 of `b` (not `a`), setting these elements to the character ‘j’. Now print `a` and `b` again. Comment on your results. Note: some insight into what’s going on may be gained from printing the addresses of elements in the arrays using the `&` operator, *eg:*

   ```
    printf("&b[i]: 0x%x\n", i, &(b[i]) );
   ```

4. **Triangles!   (4 pt.)**

   Write a program to create the following output:

   ```
             *
            ***
           *****
          *******
         *********
        *         *
       ***       ***
      *****     *****
     *******   *******
    ********* *********
   ```

   Achieve this using only `for` loops, two integer variables, `putchar` and the following defines :

   ```
   #define HEIGHT 5
   #define BASE   9
   ```