Data Structure: Assignment 1

Due: 2018/10/12 (Friday)

- Requirement: Implementation with C program
- Total: 100 points (**20 points** for <u>questions 1-3</u> and **40 points** for <u>question 4</u>)

◆ Submission instructions:

[1]. Write a "**README file**" including the answers to problems 1-4, and a detailed note about the functionality of each of the above programs, and complete instructions on how to run them.

[2]. Make sure you include your name in each program and in the README file. **Each question has one C program file and one README file.** Make sure all your programs are fully commented, and compile and run correctly on the Linux-based machines.

[3]. Submit your assignment to the portal system by the due date.

---

1. Write a c program to read from a file that contains a list of floating numbers, calculate their summation, and print out the summation. Users specify the input file name.

   For example,

   In SumFloatingt.txt, there are *n* (from 0 to MAX_SIZE=4096) floating numbers: "232 2.3 44.5 22"

   You should output 300.8 in the terminal screen

2. Implement a data structure that returns the larger one of two given values. Test your template on int, float, and string.

3. Write a program to read from standard input a list integer numbers and output the maximum value in the input list.
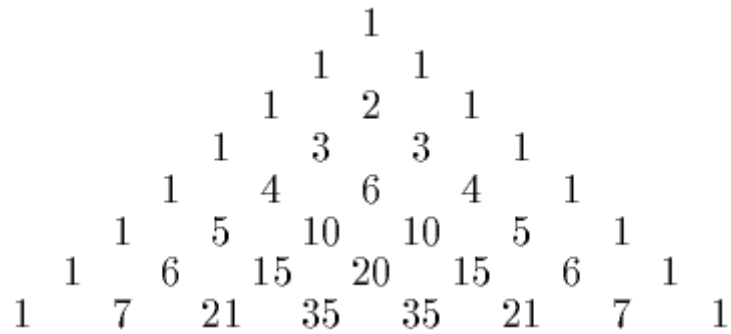
   For example,

   Test numbers from the standard input:

   1 34 2 34 12 67 22

   You program should output:

   67

4. Pascal's Triangle is an infinite triangular array of integers that starts out as shown below:

```
                    1
                1       1
            1       2       1
        1       3       3       1
    1       4       6       4       1
1       5      10      10       5       1
1   6      15      20      15       6       1
1   7   21      35      35      21      7       1
```

The number in row n and column m of the triangle is denoted by P(*n*, *m*).

If the number is on the left edge or right edge of the triangle, it is 1. That is:

$$P(n, m) = 1, \text{ if } m=1 \text{ or } m=n$$

Otherwise, it is the sum of the two numbers closest to it in the preceding row.
That is:

$$P(n, m) = P(n\text{-}1, m\text{-}1) + P(n\text{-}1, m)$$

- Write a program that takes (*n*, *m*) as inputs and outputs p(*n*, *m*).
- Use an array (or arrays) to implement the program.
- Describe how you design the program in the readme file.
- Explain what time/space complexity is in the readme file.
- Analyse the time/space complexity of your program in the readme file.