Programming Assignment #1

Lecturer: Prof. Seung-Hwan Baek
Teaching Assistants: Yujin Jeon, Suhyun Shin, Juhyung Choi, Eunsue Choi

***** PLEASE READ THIS GRAY BOX CAREFULLY BEFORE STARTING THE ASSIGNMENT *****

Due date: 11:59PM March 19, 2024

Evaluation policy:

- Late submission penalty
  - 11:59PM March 19 ~ 11:59PM March 20
    - Late submission penalty (30%) will be applied to the total score.
  - After 11:59PM March 20:
    - 100% penalty is applied for that submission.
- Your code will be automatically tested using an evaluation program.
  - Each problem has the maximum score.
  - A score will be assigned based on the behavior of the program.
  - Full points will be awarded only if all test cases are passed; there are no partial points.
  - Points will be deducted for any typos or incorrect formatting.
- We won't accept any submission via email - it will be ignored.
- Do not modify auxiliary files.
  - Such as: utils.h/cpp, evaluate.cpp, and so on.
- Compile your file(s) using 'Replit' or 'CLion' and check your program before the submission.
- All characters in submit.cpp should be in uppercase letters, e.g., 'TURE','FALSE',(except for [Task1], [Task2])
- Please do not use C++ standard template library.
  - Such as:
    - #include <queue>
    - #include <vector>
    - #include <stack>
  - Any submission using STL library will be disregarded.

File(s) you need to submit:

- pa1.cpp (Do not change the filename!)

Any questions? Please use PLMS - Q&A board in English.

0. Basic instruction

Please refer to the instruction document, “DataStructure_PA_instructions.pdf”.

```
>> g++ -std=c++11 -o pa1.exe pa1.cpp utils.cpp
```

1. Asymptotic analysis (1 pts)

a. Choose the time complexity of the following **factorial** function.

b. Factorial

- Input : An integer n>=1
- Output : The factorial of n(n!), which is the product of all integers from 1 to n

```
int factorial(int n){
        int f[n+1];
        f[0] = 1;
        for(int i=1; i<=n; i++){
           f[i] = i * f[i-1];
        }
        return f[n];
   }
```

1. O(1)
2. O(n)
3. O(log(n))
4. O(n^2)

c. Example output : If you choose O(1), then print 1

```
>> ./pa1.exe 1
[Task 1]
1
```

2. Asymptotic analysis (1 pts)

a. Choose the time complexity of the following **sumOfPrimes** function.

b. sumOfPrimes
   - Input : An integer n >= 2
   - Output : The sum of all primes up to and including n

```
int isPrime(int number) {
    if(number <= 1) return 0;
    for (int i = 2; i * i <= number; i++){
        if(number%i==0) return 0;
    }
    return 1;
}

int sumOfPrimes(int n){
    int sum = 0;
    for (int i = 2; i <= n; i++){
        if(isPrime(i)){
            sum+=i;
        }
    }
    return sum;
}
```

1. O(log(n))
2. O(nlog(n))
3. O(n^1.5)
4. O(n^2)

c. Example output : If you choose O(log(n)), then print 1

```
>> ./pa1.exe 2
[Task 2]
1
```

3. Application of List (4 pts)

a. Implement a function that can insert or delete an integer into the list. A user can insert an element in descending order or delete an element at the specific index.

If the specified index is out of range of the given list, print "ERROR".

b. Input & Output

Input : Sequence of commands, which is one of the following,

- ('insert', integer value) : insert integer value at the appropriate position in the list, while ensuring that the elements within the list are always in descending order.
- ('delete', index) : delete an element at the index in the list. Index indicates zero-based index.
- Array index indicates zero-based index.

Output :

- After inserting or deleting elements in an array, the resulting string should have spaces between each element, *but no space after the last element*.
- *"ERROR" if the index is out of range.*

c. Example Input & Output

Input	Output
[('insert',1), ('insert',2), ('insert',3), ('delete',1)]	3 1
[('delete',0)]	ERROR
[('insert',3), ('delete',2)]	ERROR
[('insert', 1), ('insert', 1), ('insert', 2), ('insert', 3), ('delete', 1)]	3 1 1

d. Example execution

```
>> ./pa1.exe 3 "[('insert',3), ('insert',5), ('insert',4), ('delete',1), ('insert',2), ('delete',0)]"
[Task 3]
3  2
```

4. Palindrome checker / Stack (3 pts)

a. Palindrome is a word, number, phrase, or other sequence of symbols that reads the same backwards as forwards. Here are a few examples:

- mom
- (())

- 4332334

b. Implement a stack and a function **IsPalindrome**. This function returns a Boolean indicating whether the given string is a palindrome, utilizing the stack you implemented.

c. Input & Output
   - Input: A string containing at least on alphabetical character. Input is case-insensitive (i.e., disregard differences between uppercase and lowercase letters). Non-alphabetical inputs should be ignored.
   - Output (Print) :
     - TRUE : If the given string is palindrome.
     - FALSE : If the given string is not palindrome

d. Example Input & Output

Input	Output
mom	TRUE
aNna	TRUE
levEL	TRUE
Happy	FALSE
postech	FALSE

e. Example execution

```
>> ./pa1.exe 4 "mom"
[Task 4]
TRUE
```

5. Queue (3 pts)

a. Implement a function that shows the values in a priority queue without using heap. We “enqueue” values in the queue and “dequeue” the front value (first inserted value). If “dequeue” operation is called for an empty queue, *you should print “EMPTY”*.

- We set the max size of queue as 100 (check the code) and we will not test over the maximum size. You do not need exceptions for this situation.

b. Input & Output (integer value range : 1 to 99)

Input: Sequence of commands, which is one of the following,

- ('enqueue',integer): enqueue integer into the current queue
- ('dequeue',0): dequeue the front value from the current queue.
- ('peek',0): print the front value
- ('show',0): show all values in queue (ex : value) in *descending* order of priority (check the example below). If empty, print nothing.

Output

- Example for *descending* order.
- Input : [('enqueue',5), ('enqueue',2), ('enqueue',3), ('show', 0)]
- Output :
- 5
- 2
- 3

c. Example input & output

Input	Output
[('enqueue',5), ('enqueue',3), ('dequeue',0), ('show', 0)]	3
[('enqueue',5), ('enqueue',3), ('peek', 0)]	5
[('dequeue',0), ('show', 0), ('dequeue',0), ('peek', 0)]	EMPTY EMPTY
[('enqueue',5), ('enqueue',3), ('enqueue',8), ('enqueue',11), ('show', 0), ('dequeue',0), ('show', 0), ('peek', 0)]	5 3 8 11 3 8 11 3

d. Example execution

```
>> ./pa1.exe 5 "[('enqueue',5), ('enqueue',3), ('dequeue',0), ('show', 0)]"
[Task 5]
3
```

6. Priority Queue (3 pts)

a. Implement a function that shows the values in a priority queue without using heap. We "enqueue" values in the queue with specific priorities *(priority is an integer starting from 1, and we define 1 as the highest priority),* and "dequeue" value with the highest priority. If "dequeue" operation is called for an empty queue, *you should print "EMPTY"*.

- We set the max size of queue as 100(Check the code) and we will not test over the maximum size. You don't need exceptions for this situation.
- We will not consider values with same priority.
- You don't need to consider exceptions for inputs other than ('enqueue', integer), ('priority', integer).

b. Input & Output (integer value & priority range: 1 to 99)
Input: Sequence of commands, which is one of the following,
- ('enqueue',integer): enqueue integer into the current queue
- ('priority',integer): specify the value's priority by integer. If you command "enqueue", you **MUST** command "priority" right after it.
- ('dequeue',0): dequeue the highest priority value from the current queue.
- ('peek',0): print the highest priority value and its priority
- ('show',0): show all values in queue with its priority in as*cending* order of priority (Please check the example below). If empty, do not print anything.

Output
- Example for *ascending* order of priority.

  Input : [('enqueue',1), ('priority',3), ('enqueue',2), ('priority',2), ('enqueue',3), ('priority',6), ('show', 0)]
  Output :
  2, 2
  1, 3
  3, 6

c. Example input & output

Input	Output
[('enqueue',5), ('priority', 1), ('enqueue',3), ('priority', 3), ('dequeue',0), ('show', 0)]	3, 3
[('enqueue',5), ('priority', 2), ('enqueue',3), ('priority', 3), ('peek', 0)]	5, 2
[('dequeue',0), ('show', 0), ('dequeue',0), ('peek', 0)]	EMPTY EMPTY
[('enqueue',5), ('priority', 2), ('enqueue',3), ('priority', 4), ('enqueue',8), ('priority', 3), ('enqueue',11), ('priority', 11), ('show', 0), ('dequeue',0), ('show', 0), ('peek', 0)]	5, 2 8, 3 3, 4 11, 11 8, 3 3, 4 11, 11 8, 3

d. Example execution

```
>> ./pa1.exe 6 "[('enqueue',5), ('priority', 1), ('enqueue',3), ('priority', 3),
('dequeue',0), ('show', 0)] "
[Task 6]
3, 3
```