LAB Manual 3 DSA

1. Given an expression containing opening and closing braces, brackets, and parentheses; implement a function "**isBalanced**" to check whether the given expression is a balanced expression or not, **using your stack implementation**. For example, {[{}{}]}[()], {{}{}}, and []{}() are balanced expressions, but {()}[) and {(}) are not balanced. In your main function test your function using the given examples.

   ```
   bool isBalanced(string exp)
   ```

2. Implement a **template-based queue using a fixed-sized array**. The required member methods are:

   int size() : returns total element stored in the queue.

   bool isEmpty(): returns true if the queue is empty else false.

   T & front(): returns, but does not delete, the front element from the queue. If there is no element, return some error.

   void dequeue(): deletes the front element from the queue. If there is no element, return some error.

   Void enqueue(T const& e): inserts the element "e" at the back of the queue if there is some space available. Otherwise it returns some error.

3. A CPU can execute many processes. However, at a given time, a CPU can execute the instructions of only one process (not true for modern CPUs, nonetheless). To manage the execution of n processes, we make use of what is known as "CPU Scheduler". We will write a basic type of scheduler which will work as follows: Suppose we have 10 processes named p1, p2,….,p10. Each process has some number of instructions n1, n2, n3…..n10, respectively. Now to execute all the processes, we first execute some instructions, let's say 3 instructions of process p1, and then we will execute 3 instructions of p2, so on and so forth. After that we will restart from process p1 and execute 3 instructions of p1, then p2, then p3, so on and so forth. Then the cycle begins again. So, the processes are being executed by CPU in circular fashion. If a process has finished its instructions during this cycle, then we remove that process from this cycle. The given example below depicts the scenario:

   Suppose we have 3 processes:
   Process_id: p1
   Total_Instructions: 7

   Process_id: p2
   Total_Instructions 6

Process_id: p3
Total_Instructions 5

Scheduler Output:
3 instructions of p1 executed.
3 instructions of p2 executed.
3 instructions of p3 executed.
3 instructions of p1 executed.
3 instructions of p2 executed.
p2 has finished execution.
2 instructions of P3 executed.
p3 has finished execution.
1 instruction of p1 executed.
p1 has finished execution.

Implement your scheduler function using the queue implemented in question 2. The function will take a file name as argument. The file contains information about all the processes to be executed. A sample file is also given. The **first number** in the file tells about the number of instructions that the CPU will execute, of a process p, at one time. The **second number** tells us about the total number of processes in the file. Test your function in main with the given file.

void scheduler(string processFile);