Data Structures
CS 246 - 040
Department of Physics and Computer Science
Medgar Evers College
Exam 2

Instructions:

- **The exam requires completing a set of tasks within 50 minutes.**
- **Modify the accompanying cpp file. Write the nonprogramming tasks as comments in the file.**
- **The runtime table can be written in a spreadsheet.**
- **Submit all your work to Github in the Exam02 directory and/or as an attachment on Google classroom under the Exam02 assessment.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating and/or failing to follow any of the rules will result in an automatic zero (0) for the exam.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, PRINT YOUR NAME AND THE DATE ON YOUR SUBMISSIONS

Grading:

Section	Maximum Points	Points Earned
Fundamental	2	
Runtime	5	
Problem Solving	15	
Implementation	3	
Total	25	

Fundamentals

1. **Write ONLY what is requested.**
   a. What is the principle of a queue?
   b. If the values `[a,b,c,d,e]` are inserted into a stack in the order listed, and then, all the values are displayed and removed from the stack, what will be displayed?
   c. The typical names of the insertion and removal methods of a queue respectively are?
   d. What is the main characteristic of a circular doubly linked list?

Runtime

2. **Construct the runtime table that includes a statement column and determine the runtime functions of the following function for the worst-case scenario. Let the cost of every operation be 1. Write the function in terms of *n*, which is the size of the array. You may need to use the ceiling or floor function for an accurate solution.**

```
bool M(const Array<int>& data)
{
  bool found = false;

  for(int i = 1;i < data.Size();i += 1)
  {
    if(data[i-1] == data[i])
    {
      if(found == true)
      {
        return true;
      }
      found = true;
    }
    else if(found == true)
    {
      found = false;
    }
  }
  return false;
}
```

Problem Solving

3. **Write the bool function `EM()` that is an equivalent linked list version of the function `M()` above.**
   Hint: Remember you need to deal with the possibility of an empty linked list.

4. **Write the definition of a bool function named `SecondToLastRemoval()` whose header is**

```
template <typename T>
bool SecondToLastRemoval(Node<T>*& root)
```

   Given that *root* points to the head of a linked list, if the linked list has at least three elements, the function removes the node before the last node in the list, and returns true; otherwise, it just returns false.

5. **Write the definition of the bool function named `IsValid()` whose header is**

```
bool IsValid(string str)
```

   It returns true if *str* is empty or represents a valid enclosure of parentheses, `()`, and square braces, `[]`. For instance, the callers `IsValid("[([])]")` and `IsValid("[(])")` will return `true` and `false` respectively.

Implementation

6. **Write a generic class named *NewStack* that contains:**
   - **private string *Stack* field named *values*.**
   - **public void method named `Push()` that takes a string parameter. It adds the parameter to *values*.**
   - **public string method named `Pop()` that takes no parameters. If *values* is empty, it returns an empty string; otherwise, it returns and removes the top item of *values*.**
   - **public bool constant method named `IsEmpty()` that takes no parameters. It returns true if *values* is empty; otherwise, it returns false.**