Stack

Objective

- To gain understanding of the implementation of STACK data structure with all basic operations.

Tasks

1. Build array based STACK using Java generic type, complete the code below
2. Build linked list based STACK using Java generic type, complete the code below
3. Implement in a method to check parenthesis validity in the given expression using array based implementation.

1. Array based stack

```
==================
public class ArrayStack<T extends Comparable<T>> {
  T stackList[];
  int top;

// constructor
  ArrayStack(int size){
   stackList=(T[]) new Comparable[size];
}
// methods
  Public void PUSH(T c)  {…}
  Public T POP()    {…}
  Public T PEEK() {…}
  Public Boolean isEmpty()  {…}
  Public Boolean isFull(){…}
}
```

2. Linked List based stack

```
====================
class StackNode<T>  {
  T info;
  StackNode<T> next;
  //Constructor
  StackNode(T data){
      Info=data;
 }}
Class LinkedStack<T>{
  StackNode<T> top;

// methods
  Public void PUSH(T c){…}
  Public T POP()  {…}
  Public T PEEK() {…}
  Public Boolean isEmpty() {…}
}
```

3. Parenthesis validation check

```
Public Boolean validate(String Exp){
Create stack s.
while (we have not read the entire string){
       symb=Read a character of the string;
       If (symb == '(' || symb == '{' || symb == '['
            s.Push (symb);
       If (symb == ')' || symb == '}' || symb == ']')    {
             if (s.empty(s))      return false;
           else{
                  item=s.peak();
                   if (item is the matching operand of symb)    // open and close parenthesis pair check
                         s.pop();
                   else{
                       return false;
                   } } }
  }//end of while
      if(stack is not empty )   return false;
       else                    return true;
```