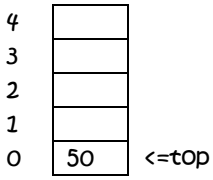
STACK DATA STRUCTURE (TUTORIAL 1)

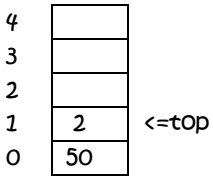
Question 1

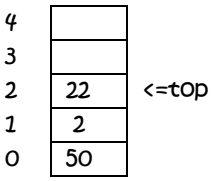
a) Consider the following Stack and draw the Stack frames after executing each statement given below.

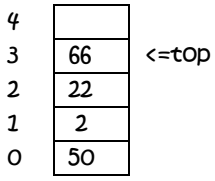
```
int a = 22, b = 44;
```

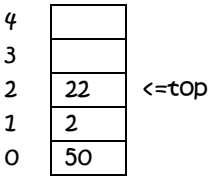
i) theStack.push(2);
ii) theStack.push(a);
iii) theStack.push(a + b);
iv) theStack.pop();
v) theStack.push(b);
vi) theStack.push(a –b);

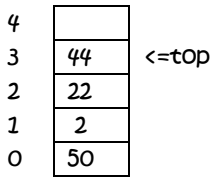


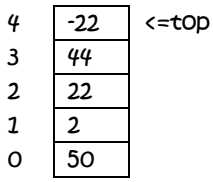












Question 2

i) Implement isEmpty() and isFull() methods of the stack class.

```
public boolean isEmpty() {
        return (top == -1);
}
public boolean isFull() {
        return (top == maxSize - 1);
}
```

ii) A stack class has already been implemented with push() , pop() and peek() methods. It is used to store characters. Write a code segment to insert following characters to a 'myStack' object created from the stack class. 'g' , 't', 'o', 'p'

```
myStack.push('g');
myStack.push('t');
myStack.push('o');
myStack.push('p');
```

iii) Write code segment to display all the values in a stack by removing them.

```
while(!myStack.isEmpty()) {
        System.out.println(myStack.pop());
}
```

iv) What is the result of section iii) above?

p
o
g
t
(and empty stack)

Question 3

A stack class called StackX has been created to store characters. 'push' and 'pop' methods have been implemented. Implement the peek method using push and pop methods.

```
public void peek() {
        char value = myStack.pop();             //remove top element
        System.out.println(value);              //print removed value
        myStack.push(value);                    //insert the value back
}
```

Additional Exercises:

Question 1

i) Implement a class called StackX to store a set of characters.

ii) Create a class called Reverser to reverse a given string using the stack class created above.

```
Class Reverser  {
private String input;
private String output;
……..
}
```

(Hint: Pass the string to be reversed as an argument to the constructor and store it in input)

iii) In main() get a string from the user and reverse the string using the Reverser class.

```
class StackX {
	private int maxSize;				//char array size
	private char[] stackArray;			//char array definition
	private int top;				//top index definition

	public StackX(String s) {			//constructor
		maxSize = s.length();			//assign String length
		stackArray = new char[s.length()];//array implementation, array size will be string length
		top = -1;
	}

	public void push(char j) {			//push method
		if(top == maxSize-1) {
			System.out.println("Stack is full");
		}
		else
			stackArray[++top] = j;		//add character to the char array
	}

	public char pop() {				//pop method
		if(top == -1) {
			System.out.println("Stack is empty");
			return 0;
		}
		else
			return stackArray[top--];	//return removed element of the stack
	}

	public boolean isEmpty() {
		return (top == -1);
	}

}  //end of class StackX


class Reverser {
	private String input;
	private String output;
	private StackX stx;			//StackX class object variable
	private char[] ch;			//char array to store return values of pop method

	public Reverser(StackX sx, String s) {	//constructor
		this.stx = sx;			//assign StackX object from Main class
		this.input = s;			//assign String value from Main class
		ch = new char[s.length()];	//array implementation
	}
```

```
	public String getOutput() {                          //method to return reversed stack array
		for(int i = 0; i < input.length(); i++) {
			ch[i] = stx.pop();  //pop method calling and assign return values to char array
			output = new String(ch);                     //convert char array to String
			//or output = String.valueOf(ch);
		}
		return output;                                   //return String
	}

}//end of class Reverser

import java.util.Scanner;                            //import Scanner class
class Main{
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);

		System.out.print("Enter a String: ");
		String str = sc.nextLine();

		StackX sX = new StackX(str);                 //StackX class instantiation
		for(int i = 0; i < str.length(); i++) {
			sX.push(str.charAt(i));    //convert string to char and pass to push method
		}

		Reverser rev = new Reverser(sX, str);  //Reverser class instantiation
		System.out.println("Reversed String is: " + rev.getOutput()); //getOutput method calling
	}
}
```

Question 2

Use the stack class created in Question1 (i) and check whether a user entered expression is correctly parenthesized.

Ex: 3 + ((6 * 2) – 3) - valid
5 * 6 + (2 - 5 - not valid

```
class StackX {
	private int maxSize;                             //char array size
	private char[] stackArray;                       //char array definition
	private int top;                                 //top index definition

	public StackX(String s) {                        //constructor
		maxSize = s.length();                        //assign String length
		stackArray = new char[s.length()];           //array size will be string length
		top = -1;
	}

	public void push(char j) {                       //push method
		if(top == maxSize-1) {
			System.out.println("Stack in overflow state");
		}
		else
			stackArray[++top] = j;                   //add character to the char array
	}

	public char pop() {                              //pop method
		if(top == -1) {
			System.out.println("Stack in underflow state");
			return 0;
		}
		else
			return stackArray[top--];                //return removed element of the stack
	}
```

```
	public char peek() {                                //peek method
		if(top == -1) {
			System.out.println("Stack is empty");
			return 0;
		}
		else
			return stackArray[top];          //return top element of the stack
	}

	public boolean isEmpty() {
		return (top == -1);
	}

}  //end of class StackX

class BalanceCheck {
	private String str;
	private StackX sX;

	public BalanceCheck(StackX s, String st) {
		this.sX = s;
		this.str = st;
	}

	public boolean getResult() {
		for(int i = 0; i < str.length(); i++) {
			if(str.charAt(i) == '(' || str.charAt(i) == '{' || str.charAt(i) == '[') {
				sX.push(str.charAt(i));
			}	//add open braces to the stack

			else if(str.charAt(i) == ')' || str.charAt(i) == '}' || str.charAt(i) == ']') {

				if(str.charAt(i) == ')' && sX.peek() == '(')
					sX.pop();
				if(str.charAt(i) == '}' && sX.peek() == '{')
					sX.pop();
				if(str.charAt(i) == ']' && sX.peek() == '[')
					sX.pop();
			}	//check peek element is the open brace of the current closing brace
		}//for loop

		return sX.isEmpty();	//stack should be empty if all the corresponding open braces
are removed

	}//getResult ends
}//end of class BalanceCheck

import java.util.Scanner;                           //import Scanner class
class PAR{
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter an expression: ");
		String str = sc.nextLine();
		StackX sX = new StackX(str);
		BalanceCheck bC = new BalanceCheck(sX, str);
		if(bC.getResult())
			System.out.println("Valid");
		else
			System.out.println("Invalid");

	}//main method
	}//class
```