**Assignment-4**

**1.Write a java program to create an user defined exception called PayOutOfBoundsException. This exception is thrown when basicpay is not in between 10000 and 30000.**

import java.util.Scanner;

class JavaException{

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

try{

if(n<10000 || n>30000)

throw new PayOutOfBoundsException();

}

catch(PayOutOfBoundsException e){

System.out.println(e) ;

}

}

}

class PayOutOfBoundsException extends Exception{

public String toString(){

return ("Invalid Pay") ;

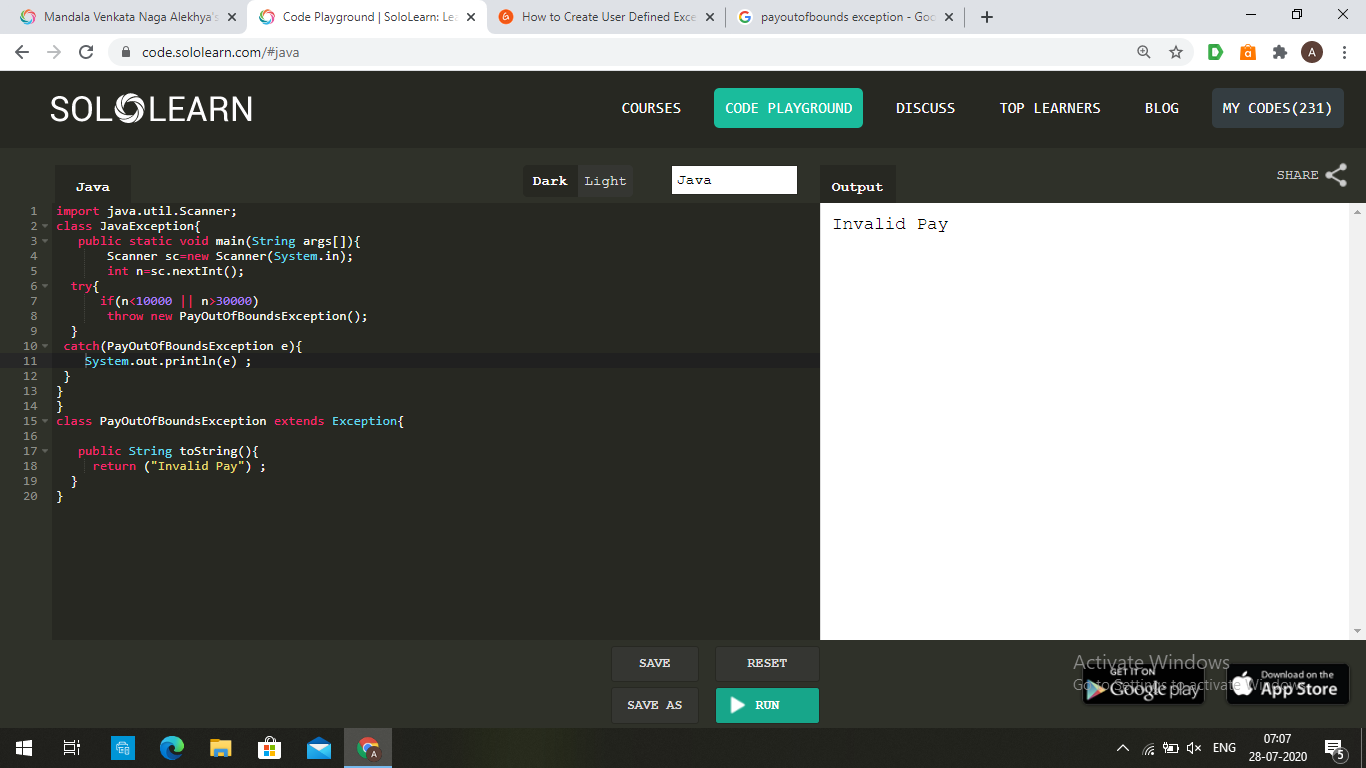
}

}

Input:

35000

Output:



**2.Write a java program to create two threads which display a message every half second.**

class clock1 extends Thread

{

clock1(String s)

{

super(s);

}

public void run()

{

for(int i=0;i <10;i++) {

String test = Thread.currentThread().getName();

System.out.println(test+" :- Good Morning");

try

{

Thread.currentThread().sleep(500);

}

catch(InterruptedException e)

{}

}

}

}

class clock2 extends Thread

{

clock2(String s)

{

super(s);

}

public void run()

{

for(int i=0;i <10;i++) {

String test = Thread.currentThread().getName();

System.out.println(test+":- Good Afternoon");

try

{

Thread.currentThread().sleep(500);

}

catch(InterruptedException e)

{}

}

}

}

class Fileread

{

public static void main(String args[])

{

System.out.println("\nThrean name:"+Thread.currentThread().getName());

System.out.println("\n");

clock1 obj1=new clock1("Thread 1");

clock2 obj2=new clock2("Thread 2");

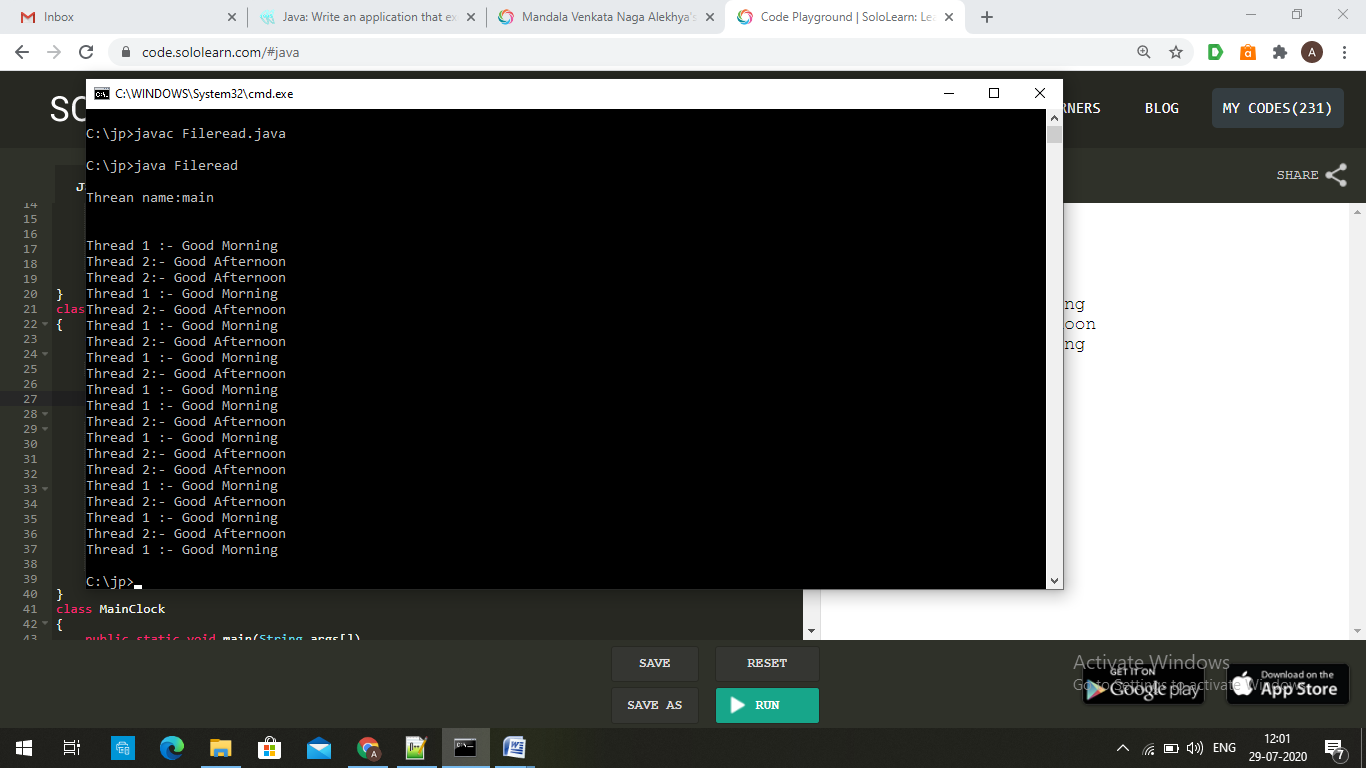
obj1.start();

obj2.start();

}

}

Output:



**3.Write a java program to implement interthread communication.**

class Customer{

int amount=10000;

synchronized void withdraw(int amount){

System.out.println("going to withdraw");

if(this.amount<amount){

System.out.println("Less balance; waiting for deposit");

try{wait();}catch(Exception e){}

}

this.amount-=amount;

System.out.println("withdraw completed");

}

synchronized void deposit(int amount){

System.out.println("going to deposit");

this.amount+=amount;

System.out.println("deposit completed");

notify();

}

}

class Test{

public static void main(String args[]){

final Customer c=new Customer();

new Thread(){

public void run(){c.withdraw(15000);}

}.start();

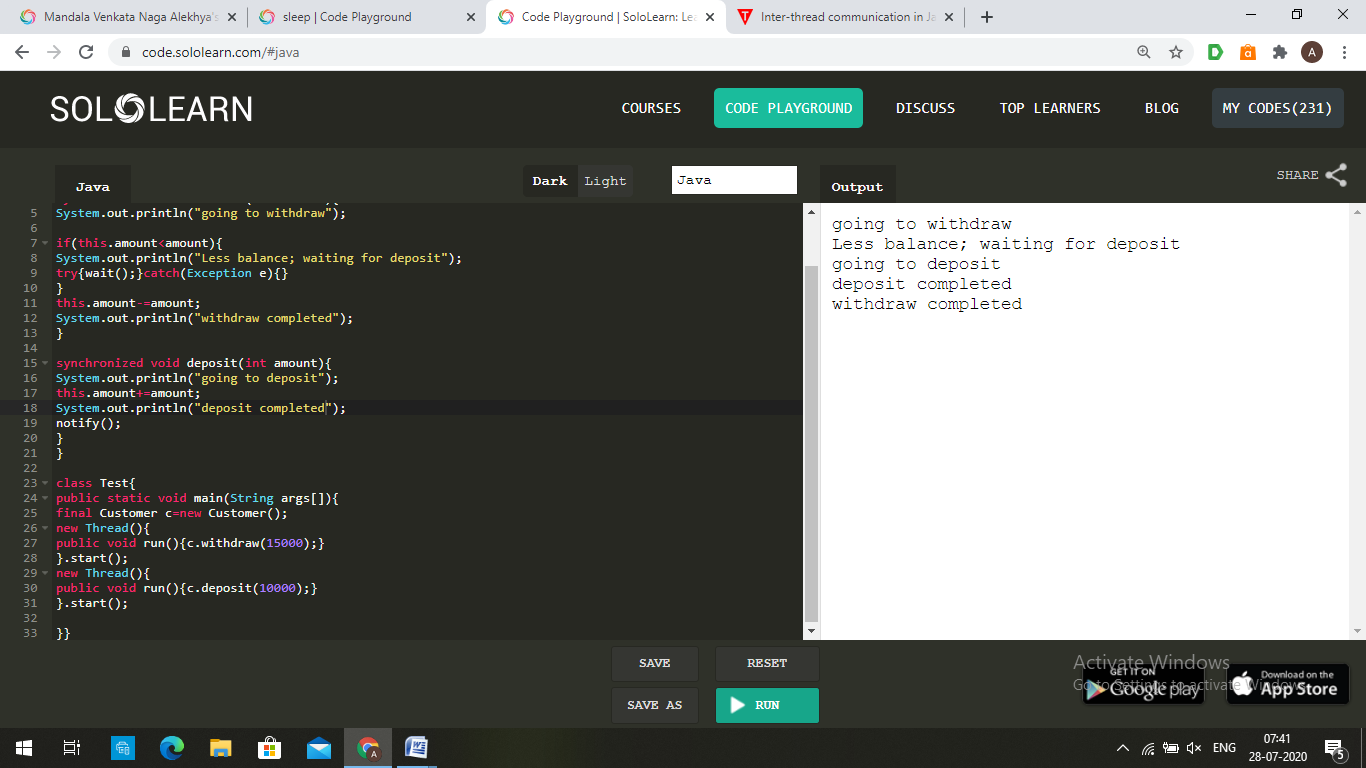
new Thread(){

public void run(){c.deposit(10000);}

}.start();

}}

Output:



**4.Write a java program to implement Thread Synchronization.**

import java.lang.\*;

class Trail extends Thread{

synchronized void show(String s){

System.out.print("Hello ");

try{

Thread.sleep(500);

}

catch(Exception e){

}

System.out.println(s+" world.");

}

}

class A extends Thread{

Trail t;

A(Trail t1)

{

t=t1;

}

public void run(){

t.show("c++");

}

}

class B extends Thread{

Trail t;

B(Trail t1)

{

t=t1;

}

public void run(){

t.show("c");

}

}

class test{

public static void main(String args[])

{

Trail t2=new Trail();

A a1=new A(t2);

B b1=new B(t2);

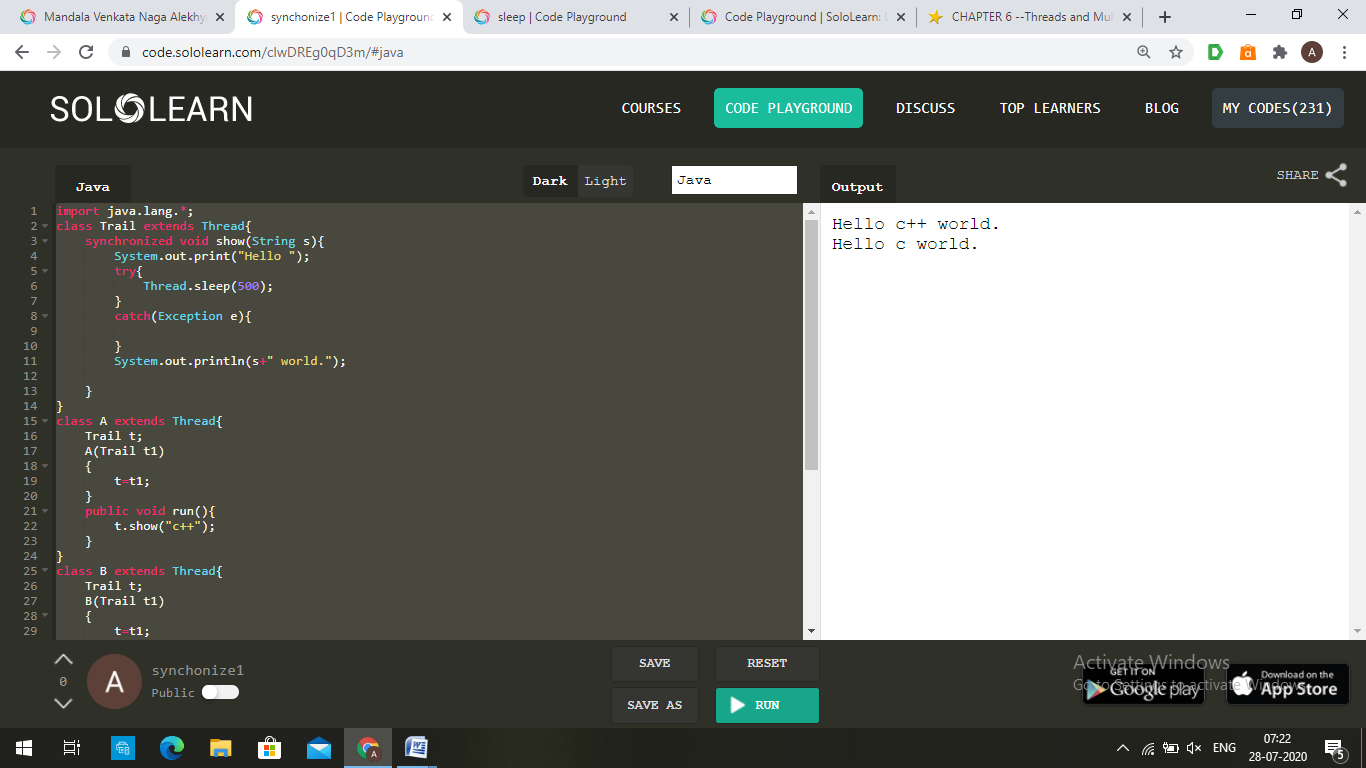
a1.start();

b1.start();

}

}

Output:



**5.Write a java program to implement Generic Class,Generic Method and Generic Constructor.**

import java.util.ArrayList;

import java.util.Arrays;

class Test<T>

{

T obj;

<T1> Test(T1 ob1) { obj = (T)ob1; }

public <T2> void getObject(T2 e) { System.out.println(obj+" "+e); }

}

class Main

{

public static void main (String[] args)

{

Test <Integer> iObj = new Test<Integer>(15);

iObj.getObject("ab");

Test <String> sObj =

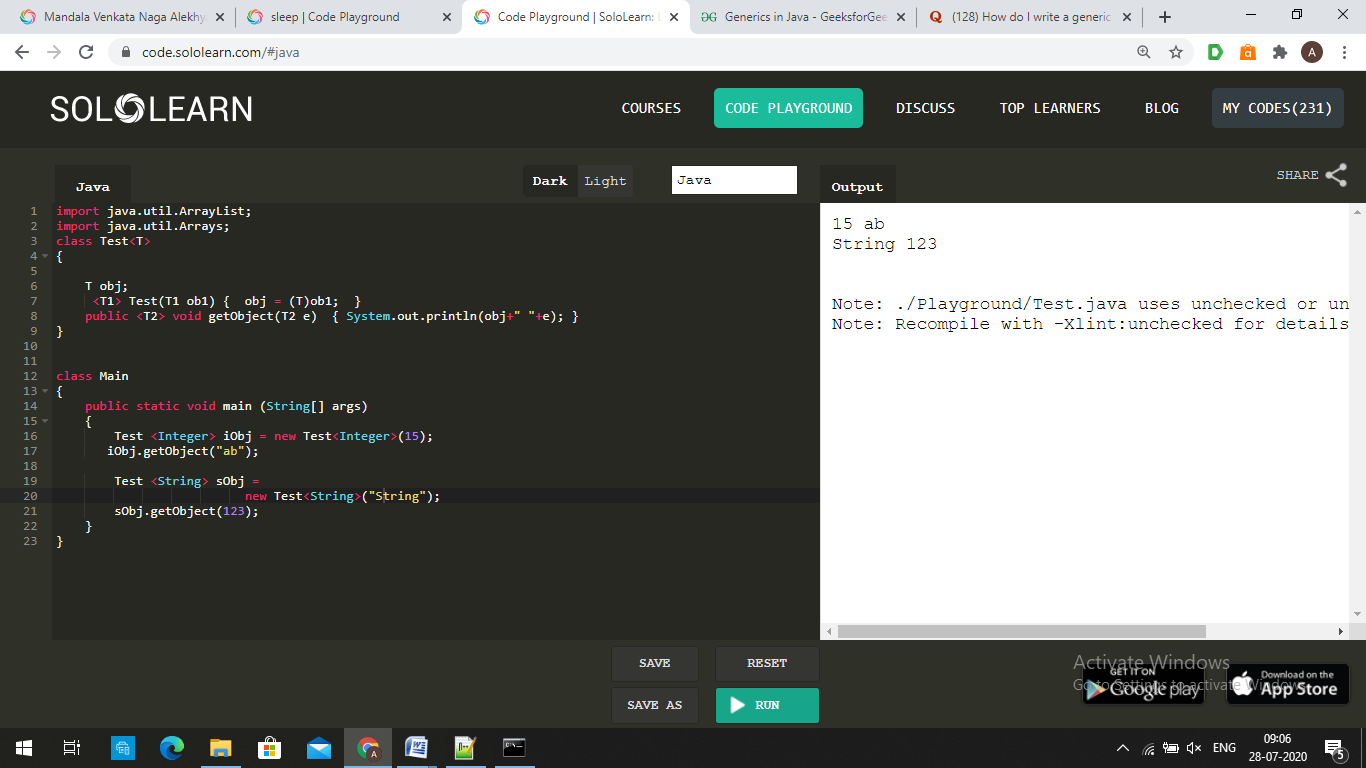
new Test<String>("String");

sObj.getObject(123);

}

}

Output:



**6.Write a java program to count no of vowels in a given file.**

import java.io.\*;

public class Fileread {

public static void main(String args[]) throws IOException {

FileReader in = null;

try {

in = new FileReader("input.txt");

int c,count=0;

char ch;

while ((c = in.read()) != -1) {

ch=(char)c;

if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u' || ch=='A' || ch=='E' || ch=='I' || ch=='O' || ch=='U' )

count+=1;

}

System.out.print("Number of vowels : "+count);

in.close();

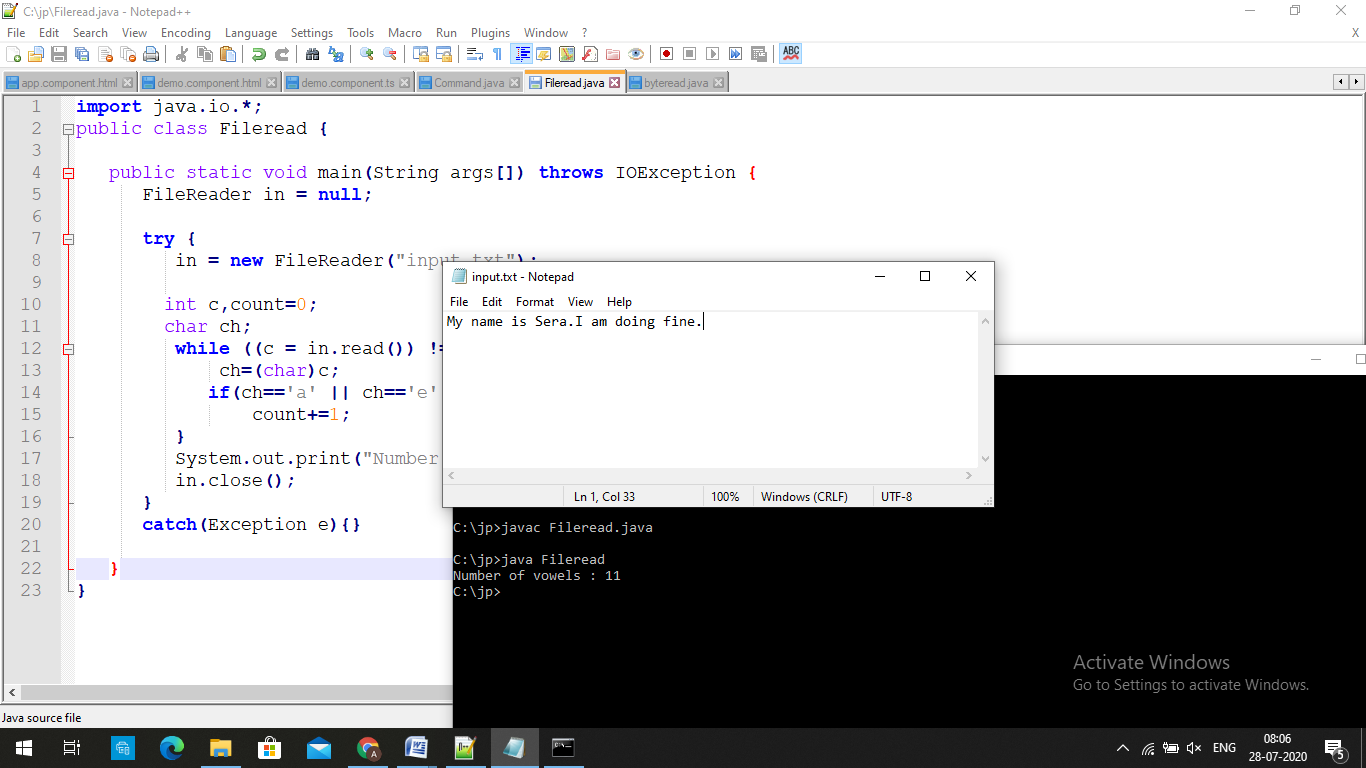
}

catch(Exception e){}

}

}

Output:



**7.Write a java program to implement autoboxing and unboxing.**

class Box

{

public static void main (String[] args)

{

Integer i = new Integer(20); //unboxing

int i1 = i;

System.out.println("Value of i: " + i);

System.out.println("Value of i1: " + i1);

int bo = 10; //autoboxing

Integer ch = new Integer(bo);

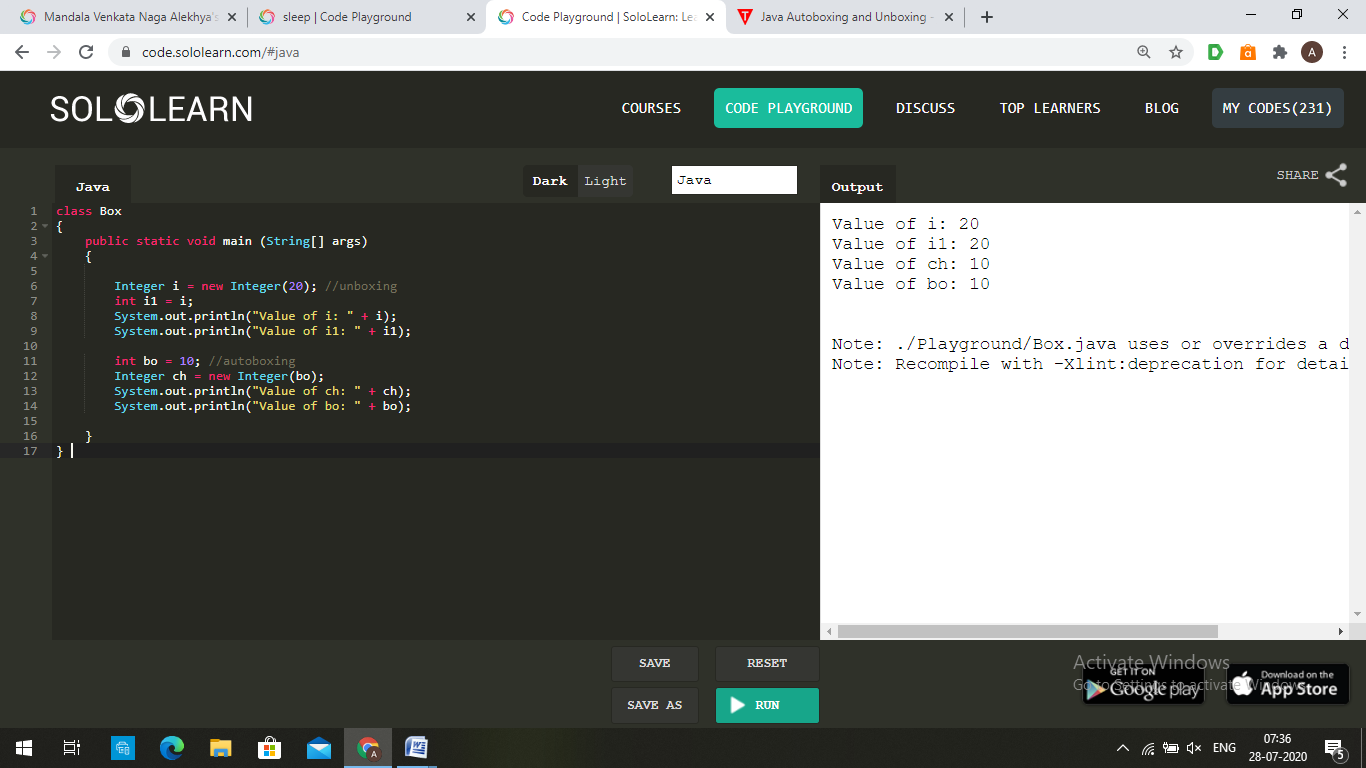
System.out.println("Value of ch: " + ch);

System.out.println("Value of bo: " + bo);

}

}

Output:



**8.Write a java program to copy a file.**

import java.io.\*;

class Fileread

{

public static void main(String args[])

throws FileNotFoundException,IOException

{

FileInputStream fis = new FileInputStream(args[0]);

FileOutputStream fos = new FileOutputStream(args[1]);

int b;

while ((b=fis.read()) != -1)

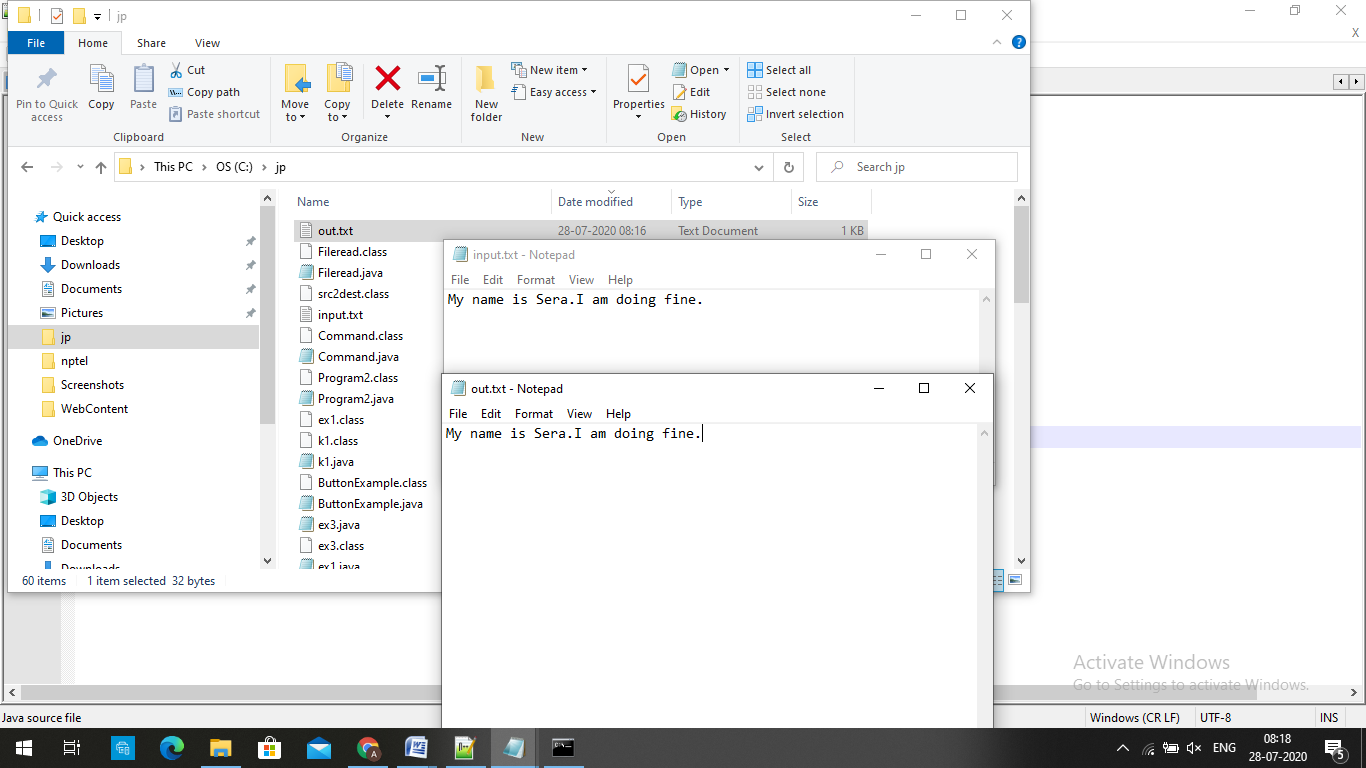
fos.write(b);

fis.close();

fos.close();

}

}



**9.Write a java program to implement Stack using Generic class**

import java.util.\*;

public class GenericStack <T> {

private ArrayList<T> stack = new ArrayList<T> ();

private int top = 0;

public int size () { return top; }

public void push (T item) {

stack.add (top++, item);

}

public T pop () {

return stack.remove (--top);

}

public static void main (String[] args) {

GenericStack<Integer> s = new GenericStack<Integer> ();

s.push (6);

int s1=s.size();

System.out.println("size of stack: "+s1);

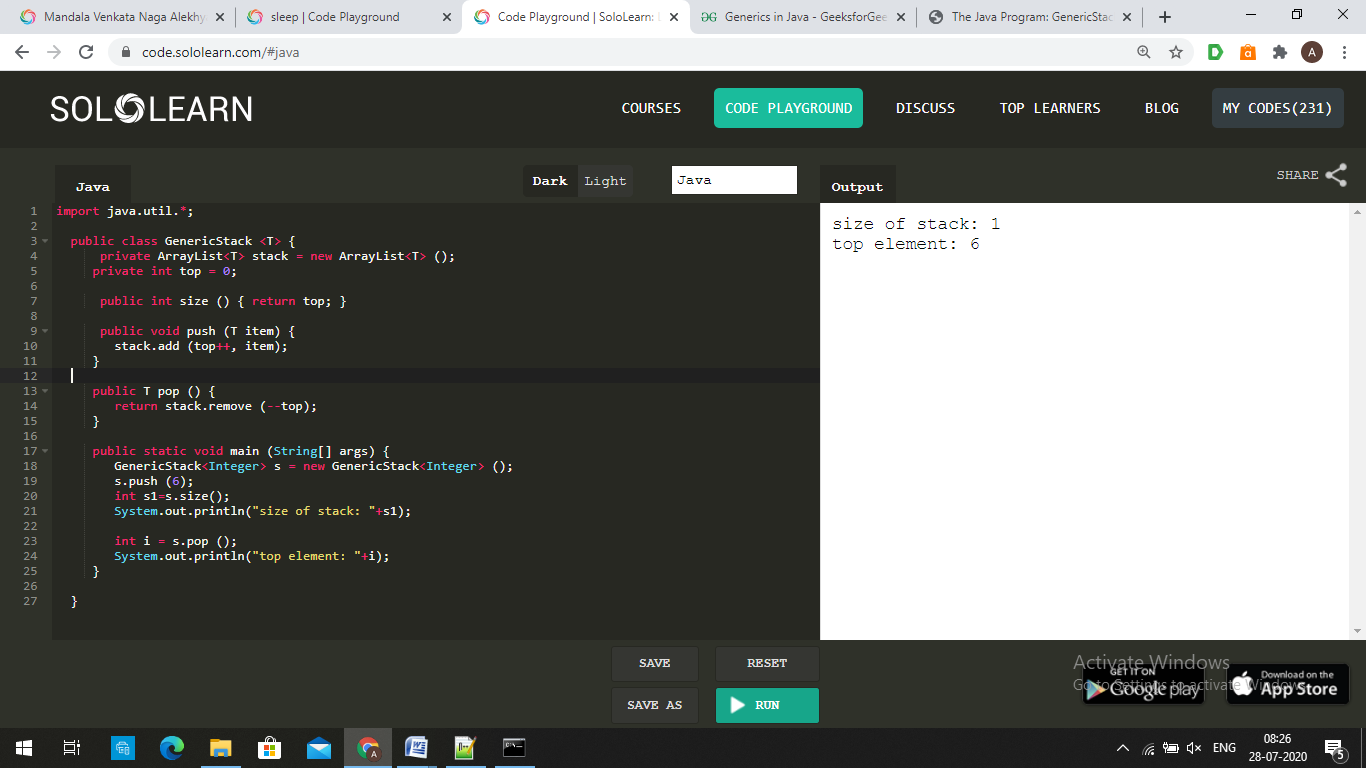
int i = s.pop ();

System.out.println("top element: "+i);

}

}

Output:



**10.Write java program to swap two values using generic method.**

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class Main {

public static final <T> void swap (T[] a, int i, int j) {

T t = a[i];

a[i] = a[j];

a[j] = t;

}

public static final <T> void swap (List<T> l, int i, int j) {

Collections.<T>swap(l, i, j);

}

private static void test() {

String [] a = {"Hello", "Goodbye"};

System.out.println("original array : "+Arrays.toString(a));

swap(a, 0, 1);

System.out.println("Swap using array : "+Arrays.toString(a));

List<String> l = new ArrayList<String>(Arrays.asList(a));

swap(l, 0, 1);

System.out.println("Swap again Using list : "+l);

}

public static void main(String args[])

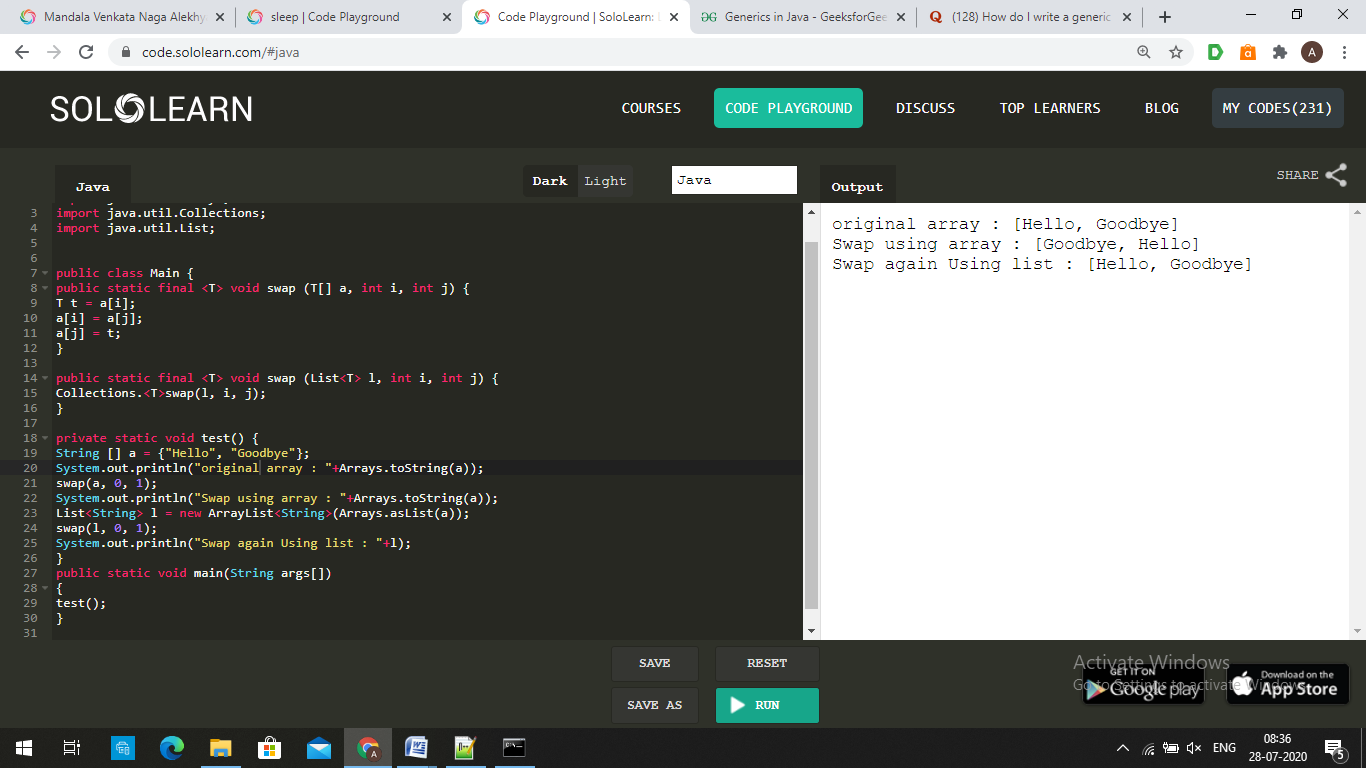
{

test();

}

}

Output:



Theory

**1. What is thread?**

A multithreaded program contains two or more parts that can run concurrently.Each part of such a program is called thread,and each thread defines a separate path of execution.

**2. Write the difference between multithreading and multitasking**

|  |  |
| --- | --- |
| Multithreading | Multitasking |
| 1.It is a process of executing multiple threads(sub-process). | 1.It is a process of executing many processes running simultaneously. |
| 2.Thread is basically a lightweight sub-process. | 2.It is basically a heavy weight process. |
| 3.In multithreading, processes are allocated same memory. | 3.In multitasking, the processes share separate memory. |
| 4.multithreading component does not involve multiprocessing. | 4.Multitasking component involves multiprocessing. |

**3. What is Enumeration?**

An Enumeration is a list of named contants.In java ,an enumeration defines a class type. An enumeration can have constructors,methods,and instance variables.

**4. What is autoboxing?**

Autoboxing is the automatic conversion that the java compiler makes between the primitive types and their corresponding object wrapper classes.For example,converting an int to an Integer, a double to a Double..

**5. What is wrapper class?**

A wrapper class is a class whose object wraps or contains a primitive datatypes.The wrapper class in java provides the mechanism to convert primitive into object and object into primitive.

**6. what is transient modifier?**

Transient in Java is used to indicate that a field should not be part of the serialization process.The modifier Transient can be applied to member variables of a class to turn off serialization on these member variables. Every field that is marked as transient will not be serialized.

**7. What is Generic class?Write the syntax of generic class.**

A generic type is a generic class or interface that is parameterized over types.We use angle brackets (<>) to specify the type parameter.

Syntax:

public class ClassName<T>

{//Class Members….}

**8. What is stream?**

A **stream** is a sequence of objects that supports various methods which can be pipelined to produce the desired result. A **stream** is not a data structure instead it takes input from the Collections, Arrays or I/O channels.

**9.What is predefined stream?**

java provides three predefined stream objects: in, out, and err, defined in the Systemclass of the java.langpackage. The out object refers to the standard outputstream or console. The in object refers to standard input, which is the keyboard. And,the err object refers to a standard error, which again is nothing but the console.

**10.What is multithreading?**

Multithreading is a process of executing multiple threads simultaneously. Multithreading is used to obtain the multitasking. It consumes less memory and gives the fast and efficient performance.

**11. What is the use of toString()?**

The toString method is used to return a string representation of an object. If any object is printed, the toString() method is internally invoked by the java compiler. Else, the user implemented or overridden toString() method is called.

**12.What is deadlock?**

Deadlock in Java is a condition where two or more threads are blocked forever, waiting for each other. This usually happens when multiple threads need the same locks but obtain them in different orders.

**13. Write inter thread communication methods.**

Inter-thread communication is a mechanism in which a thread is paused running in its critical section and another thread is allowed to enter (or lock) in the same critical section to be executed.It is implemented by following methods of Object class:

* wait()
* notify()
* notifyAll()

**14. Write the difference between Checked and Unchecked exception.**

|  |  |
| --- | --- |
| Checked Exception | Unchecked Exception |
| 1.The exceptions which are checked by the compiler for smooth execution of the program at runtime are called Checked Exception | 1.The exceptions which are not checked by the compiler are called Unchecked Exceptions |
| 2.If these are not handled properly, they will give compile time error | 2.If these exceptions are not handled properly, they don’t give compile time error. But the program will be terminated prematurely at runtime |
| 3.These occur at compile time | 3.These occur at runtime |
| 4.All subclasses of Exception class except RuntimeException have Checked Exception | 4.All subclasses of RunTimeException and Error are unchecked exceptions |

**15. What is thread synchronization?**

When we start two or more threads within a program, there may be a situation when multiple threads try to access the same resource and finally they can produce unforeseen result due to concurrency issues.So there is a need to synchronize the action of multiple threads and make sure that only one thread can access the resource at a given point in time. This is implemented using a concept called monitors.