```
//abstact class exampleimport java.util.Scanner;
abstract class Ab1
{
    public void Ab1() {
        System.out.println("regular method");
        }
            abstract void ex();


}
class Ab2  extends Ab1{
  public void ex()
  {
      System.out.println("example");
  }

}
class AbstractExp2
{
    public static void main(String []args) {
        Ab1 a=new Ab2();
        a.ex();
        a.Ab1();
    }

}
2.

import java.util.Scanner;
abstract class Abstract1
{
    abstract void met1();

}
class Normal extends Abstract1
{
public void met1()
{
    System.out.println("it was derived from absractt");
}
}
class AbstractExp1
{
    public static void main(String []args) {
        Abstract1 a1=new Normal();
        a1.met1();
    }
}

3.constructor with argument
class ConstructArgument
{
    public void ConstructArgument(int a) {
```

```
        System.out.println("this constructor contains"+ a +"as a
argument");


    }
}
class ConstructorArg
{
    public static void main(String args[]) {
        ConstructArgument ca=new ConstructArgument();
       ca.ConstructArgument(20);

    }
}


4.
class DefaultConstructor
{
    public void DefaultConstructor()
    {
        System.out.println("this consturctor have no arguments");
    }
}
class ConsturctorDef
{
    public static void main(String []args)
    {
        DefaultConstructor df=new DefaultConstructor();
        df.DefaultConstructor();

    }
}

5.
import java.util.Scanner;

class Factorial
{
    int fact=1;
public void exFact(int a)
{
    for(int i=1;i<=a;i++)

    fact*=i;
    System.out.println("he factorial is"+fact);
}
}
class ExaClass
{
    public static void main(String []args) {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        Factorial f=new Factorial();
```

```
        f.exFact(n);

    }
}
```

6.final key

```
 class FinalKey
{
   /* public final void m1()
    {
        System.out.println("Final method");
    }*/
    public static void main(String args[])
    {
       final int a=20;

        System.out.println(a);
 }
}
```

7.

```
import java.util.Arrays;
import java.util.Scanner;
class FindDupArray
{
    public static void main(String []args)
    {
        Scanner s=new Scanner(System.in);
        int n=s.nextInt();
        int a[]=new int[n];
        int count=0,dup1=0;
        for(int i=0;i<n;i++)
        {
            a[i]=s.nextInt();
        }
        for(int i=0;i<n;i++)
        {
            int dup=a[i];
            for (int j=i+1;j<n;j++)
            {
                if (dup==a[j])
                 {
                     dup1=a[j];
                     count++;
                 }
            }
        }
        if(count==0)
         System.out.println("The array doesn't contain duplicate value");
         else
         System.out.println("the duplicate value is "+dup1);
    }
```

```
}

8.
class Exa1
{

    public void pro() {
System.out.println("This is my first program");
    }
}
class FirstClass
{
    public static void main(String []args) {
        Exa1 e=new Exa1();
        e.pro();

    }
}


9.inheritance


import java.util.Scanner;
class A
{
    public void exp1(){
    System.out.println("this is parent class");
}
}
class B extends A
{
    public void exp2(){
    System.out.println("it was a child class");
}
}
class InheritanceExm
{
    public static void main(String []args) {
        //A a=new A();
      B b=new B();
        b.exp1();
        b.exp2();


    }
}

10.
class Exp1
{
    int a;
    String name;
}
```

```
class InstanceVar
{
    public static void main(String []args)
    {
        Exp1 o1=new Exp1();
        Exp1 o2=new Exp1();
        o1.a=10;
        o2.a=20;
        o1.name="madhu";
        o2.name="jan";
        System.out.println("First object");
        System.out.println(o1.a);
        System.out.println(o1.name);
        System.out.println("second object");
        System.out.println(o2.a);
        System.out.println(o2.name);
    }
}
```

11.

```
interface NewInter
{
    default void display()
    {
        System.out.println("this is default interface");
    }
}
class InterfaceDef implements NewInter
{
    public static void main(String args[])
    {
        InterfaceDef i=new InterfaceDef();
        i.display();

    }
}
```

12.

```
//import java.util;
interface FirstInterface
{
    final String str="Hello";
    public void display();
}
class ExpClass implements FirstInterface
{
    public void display()
    {
        System.out.println("This class implements interface");
    }
}
class InterfaceExp
{
    public static void main(String []args)
```

```
    {
        ExpClass e=new ExpClass();
        e.display();
        System.out.println("the String "+e.str);

    }
}
```

13.

```
class LocalVar
{    static int n=5;
    //int n=5;
    public void m1()

    {
    n=10;
        //int n=5;
        System.out.println("the value of n is "+ n);

    }
    public static void main(String args[])
    {
        LocalVar l=new LocalVar();
        l.m1();
       System.out.println("the value of n is "+ n);
    }

}
```

14.

```
public class Memory {

      public static void main(String[] args) { // Line 1
            int i=1; // Line 2
            Object obj = new Object(); // Line 3
            Memory mem = new Memory(); // Line 4
            mem.foo(obj); // Line 5
      } // Line 9

      private void foo(Object param) { // Line 6
            String str = param.toString(); //// Line 7
            System.out.println(str);
      } // Line 8

}
```

15.

```
class A
{
    public void m1()
    {
        System.out.println("It is a base class");
    }
```

```
}
class B extends A
{
    public void m1(int a)
    {
        System.out.println("It is a derived class"+ a);
    }
}
class MethodOverLoad
{
    public static void main(String args[])
    {
        B a1=new B();
        a1.m1();
        a1.m1(5);
    }
}
```

16.

```
class A
{
    public void m1()
    {
        System.out.println("base class");
    }
}
class B extends A
{
    public void m1()
    {
        System.out.println("derived1 class");
    }
}
class C extends B
{
    public void m1()
    {
        System.out.println("derived2 class");
    }
}
class MethodOverRide
{
    public static void main(String args[])
    {
        A a=new A();
        B b=new B();
        C c=new C();
        a.m1();
        b.m1();
        c.m1();
    }
}
```

17.

```
import java.util.Scanner;
class PassingArray
{
 int sum=0;
    public void met1(int arr[],int m)
    {
        for(int i=0;i<arr.length;i++){
        sum=sum+arr[i];
        }
        m=m-sum;
        System.out.println("The missing number in the array is "+m );

    }
}
class MissingArray
{
    public static void main(String []args) {

    PassingArray p=new PassingArray();

    Scanner s=new Scanner(System.in);
    int n= s.nextInt();
    int miss=(n*(n+1))/2;
    int[] arr1=new int[n];

    for(int i=0;i<arr1.length-1;i++)
    {
        arr1[i]=s.nextInt();
    }
    p.met1(arr1,miss);
    }
}
```

18.

```
interface MulInter
{
    default void display()
    {
        System.out.println("parent 1");
    }
}
interface MulInter1
{
    default void display()
    {
        System.out.println("parent 2");
    }
}
class MulInherInterface implements MulInter,MulInter1
{
    public void display()
    {
        MulInter.super.display();
        MulInter1.super.display();
```

```
    }
    public static void main(String []args)
    {
        MulInherInterface m=new MulInherInterface();
        m.display();
    }
}
```

19.

```
import java.io.*;
class RetriveChar
{
    public static void main(String []args) throws IOException
    {
        BufferedReader br= new BufferedReader(new
InputStreamReader(System.in));
        System.out.println("Enter the string");
        String name=br.readLine();
        char s=name.charAt(3);
        System.out.println(s);

    }
}
```

20.

```
class Student{
    int age;               //instance variable
    String name;     //instance variable

    public Student()
    {
        this.age = 0;
        name = "Anonymous";
    }
    public Student(int Age, String Name)
    {
        this.age = Age;
        setName(Name);
    }
    public void setName(String Name)
    {
        this.name = Name;
    }
}

 class Main{
      public static void main(String[] args) {
            Student s;                    //local variable - reference
            s = new Student(23,"Jonh");
            int noStudents = 1;           //local variable
      }
}
```

21.

```
class StaticKey
{
    static int amount=0;
    public  void m1(int a) {

        amount+=100;
        }
        public static void main(String []args) {
            StaticKey s=new StaticKey();
            s.m1(1000);
            System.out.print(amount);
        }
}
```

22.

```
class StaticVariable
{
   static int count=0;
   public void increment()
   {
       count++;
   }
   public static void main(String args[])
   {
    StaticVariable obj1=new StaticVariable();
       StaticVariable obj2=new StaticVariable();
       StaticVariable obj3=new StaticVariable();
       obj1.increment();
      obj2.increment();
      obj3.increment();
       System.out.println("Obj1: count is="+obj1.count);
       System.out.println("Obj2: count is="+obj2.count);
       System.out.println("Obj3: count is="+obj3.count);
   }
}
```

23.

```
import java.util.Scanner;
class StringRev
{
    public static void main(String []args) throws Exception {
        Scanner s=new Scanner(System.in);
        String str= s.nextLine();
        String revStr="";
        for(int i=str.length()-1;i>=0;i--)
        {
            revStr=revStr+ str.charAt(i);
        }
System.out.println(revStr);

    }
```

```
}
```

24.

```
import java.util.Arrays;
class StrSorting
{
    public String StrSorting(String inpuString)
    {
        char temp[]= inpuString.toCharArray();
        Arrays.sort(temp);
        return new String(temp);
    }
}
class StringSort{
    public static void main(String []args) {
        String a= "34521";
        StrSorting ss=new StrSorting();
        String b=ss.StrSorting(a);
        System.out.println("the string before sorting"+ a);
        System.out.println("the string after sorting "+ b);
    }
}
```

25.

```
import java.io.*;

class StrOperation1
{
    static String s1="hello world";
    static String s2="WELCOME";
    public static void main(String []args) {

        upCase();
        lowCase();
        replaceStr();

    }

   static void upCase()
        {
            System.out.println("this is upper case:" + s1.toUpperCase());

        }
        static  void lowCase()
        {
            System.out.println("this is lower case"+ s2.toLowerCase());

        }
        static void replaceStr()
        {
            System.out.println("repalced string " + s1.replace('o','e'));
        }
```

```
}
```

26.

```
import java.util.Scanner;
class StrPalindrome
{
    public static void main(String []args) throws Exception {
        Scanner s=new Scanner(System.in);
        String word=s.nextLine();
        String temp=word;
        String pal="";
        for(int i=temp.length()-1;i>=0;i--)
        {
            pal=pal+ temp.charAt(i);

        }
        if(pal==word)
         {
             System.out.println("palindrome");
         }
         else
         {
             System.out.println("not palindrome");
         }

    }
}
```

27.

```
import java.util.Scanner;
class StrVowels
{
    public static void main(String []args) {
        int vowCount=0;
   Scanner s= new Scanner(System.in);
   System.out.println("enter the String");
   String str=s.nextLine();

   //char[] vowels={'a','e','i','o','u'};
   char[] temp= str.toCharArray();
   for(char c:temp)
   {
       switch(c)
        {
            case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u':
            case 'A':
            case 'E':
            case 'I':
```

```
            case 'O':
            case 'U':
            vowCount++;
          break;
        }

       //count++;

    }
System.out.println(vowCount+" number of vowels");
System.out.println(str.length()-vowCount+" number of consonants");

    }
}
```

28.

```
class Domestic
{
     String name="cat";
    public void Domestic()
    {
        // static String name="cat";
       System.out.println("domestic animal");
    }
}
    class Dog extends Domestic
{
    public void Dog()
    {  super.Domestic();
        System.out.println(super.name);
        System.out.println("dog is created");
    }
}
    class SuperKey
    {
    public static void main(String args[]){
        Dog d=new Dog();
       d.Dog();

    }

}
```