**TASKS**

**1.Can abstract class have constructors in java?**

**A.** As we know that abstract class cannot be instantiated with new() operator or any other ways. But we can create a constructor for the abstract class either explicitly or the compiler only will add default constructor. **Chaining of constructors** is one of the reason abstract class can have constructors.

**2.Can abstract class implements interface in Java? Do they require to implement all methods?**

**A.** Yes, class can implement interface by using **Implements** keyword. Since they are abstract, they don’t need to implements al methods. If we implement all the methods, we will get the advantage of flexibility of abstract class to implement common behavior at one place.

**3.Can abstract class be final in java**

**A.** No, abstract cannot be final in Java. If we make the abstract class final, it will stop from being inherited, **Abstract** keyword tells us to extend a class declared with abstract keyword to use it. But **final** keyword prevents a class from being executed.

**4.Can abstract class have static methods in java?**

**A.** Yes, class can declare and define static methods.

**5.Can you create instance of abstract class?**

**A.** No, we cannot create instance of abstract class in Java. because we know that abstract class is incomplete. Java compiler throw error, when a code tries to instantiate abstract class.

**6.Is it necessary for abstract class to have abstract method?**

**A.** No, it’s not mandatory for an abstract class to have any abstract method. An abstract class can have the following methods.

1.All abstract methods

2.All non-abstract methods(concrete)

3.A mixture of both methods

**7.Difference between abstract class and interface in Java?**

**A)1.** Abstract class can have **abstract** and **non-abstract** methods. Interface can have only **abstract** methods. but from java 8, it can have **default** and **static** methods also.

**2.**Abstract class doesn’t support multiple inheritance, interface supports multiple inheritance.

**3.**Abstract class can have final, static, concrete method,interface can have only static and final variables.

**8.When do you favor abstract class over interface?**

**A.**as we know that multiple inheritance is not possible by extend keyword, in this situations we use the interface. If there is any default or common behavior among all the classes, we prefer abstract class.

**9.What is abstract method in java?**

**A.** Abstract method is a method without body. You just declare method, without defining it.By default methods declared in interface are abstract. if you want to implement you need to extend abstract class and override the abstract.

### 10. Can abstract class contains main method in Java?

**A.** Yes, abstract class can contain main method, it just another static method and you can execute Abstract class with main method, until you don’t create any instance.

main method, until you don’t create any instance.

**11.What is static block in java?**

**A.** Static block is used for initializing the static variables. This block gets executed when the class is loaded in the memory. A class can have multiple Static blocks, which will execute in the same sequence in which they have been written into the program.

**12.What is the need of static block?**

**A.** When you need to execute any block before main method just **use** it. It is a set of statements, which will be executed by the JVM before execution of main method.

**13.Can we overload static methods in java?**

**A.** Yes, we can overload static methods, but we can’t override a static method.

**14.What is the difference between final and static keywords?**

**A.** When we apply **final** keyword to a variable, the value of that variable remains constant.

When we apply **static** keyword to a variable,It means It belongs to class.

When we apply **static** keyword to the method, it means the method can be accessed without creating a instance of the class.

**15.Can we call super class static methods from sub class?**

**A.** Yes we can call super class static method using subclass name.super class method()

But we cannot call super class static method if the same static method is defining in the sub class by using sub class name.

**16.Write a note on covariant return type with example code.**

**A.** The covariant return type specifies that the return type may vary in the same direction as subclass. Before java 5 it is not possible to override any method by changing the return type. But it is possible from java5. It helps in preventing run-time Class Cast Exceptions on returns .

EXAMPLE :

class A { }

class B extends A { }

class Base {

A method1()

{

System.out.println(“Base method1()”);

Return new A();

}

}

Class Derived extends Base

{

B method1()

{

System.out.println(“Derived method1()”);

Return new B ();

}

}

Public class CovariantExample{

Public static void main(String args[])

{

Base b=new Base();

b.method1();

Derived d=new Derived();

d.method1 ();

}

}

**17.Write a note on Enum with example code.**

**A.** An enum is a special type of datatype which is basically a set of constants. It can be used for days of the week,directions etc. enum constants are static and final implicitly.

**enum** Day

{

    SUNDAY, MONDAY, TUESDAY, WEDNESDAY,

    THURSDAY, FRIDAY, SATURDAY;

}

public class Test

{

    Day day;

    public Test(Day day)

    {

        this.day = day;

    }

    public void dayIsLike()

    {

        switch (day)

        {

        case MONDAY:

            System.out.println("Mondays are bad.");

            break;

        case FRIDAY:

            System.out.println("Fridays are better.");

            break;

        case SATURDAY:

        case SUNDAY:

            System.out.println("Weekends are best.");

            break;

        default:

            System.out.println("Midweek days are so-so.");

            break;

        }

    }

     public static void main(String[] args)

    {

        String str = "MONDAY";

        Test t1 = new Test(Day.valueOf(str));

        t1.dayIsLike();

    }

}

**18.Write a note on use of super keyword and super() method.**

**A.**Super is the keyword in java refers to the objects of immediate parent class.

Super can be used to invoke immediate parent class method.Super() method can be used to invoke immediate parent class constructor.

**19.Write a code to implement abstraction using interface.**

**A.** Interface in java is a mechanism to achieve abstraction and multiple inheritance in Java

By using this we can achieve loose coupling.

Example:

Interface Bank{

Float rateOfInterese();

}

Class SBI implements Bank{

Public float rateOfInterest(){

return 9.15f;

}

}

Class PNB implements Bank{

Public float rateOfInterest(){

return 9.7f;

}

}

Class InterfaceExample{

Public static void main(String[] args){

Bank b = new SBI();

System.out.println(“ROI:”+b.rateOfInterest());

}

}

**20.Write a program to find the duplicate values of an array of integer values.**

**A.**

Import java.util.Arrays;

Public class DuplicateValues

{

Public static void main(String[] args)

{

Int[] arr={1,2,3,5,6,3,2,6};

For(int i=0;i<arr.length-1;i++)

{

For(int j=i+1;j<arr.length;j++)

{

If((arr[i]==arr[j])&&(i!=j))

System.out.println(“Duplicate Element : “+arr[j]);

}

}

}

}

**21.Write a java program to reverse an array of integer values.**

**A.** import java.util.Arrays;

Public class ReverseElements{

Public static void main(String[] args){

int arr[]={2,3,4,5,6,7,8,9,10};

System.out.println(“Original array:”+Arrays.toString(arr));

for(int i=0;i<arr.length/2;i++){

int temp=arr[i];

arr[i]=arr[arr.length-i-1];

arr[arr.length-i-1]=temp;

}

System.out.println(“Reverse array:”+Arrays.toString(arr));

}

}

**22.Write a program to remove a specific element from an array.**

**A.** import java.util.Arrays;

Public class RemoveElements{

Public static void main(String[] args){

int arr[]={25,65,24,89,32,64,75,98};

System.out.println(“Original array:”+Arrays.toString(arr));

Int removeIndex=1;

For(int i=removeIndex;i<arr.length-1;i++){

arr[i]=arr[i+1];

}

System.out.println(“After removing the elements from array:”+Arrays.toString(arr));

}

}

**23.Write a java program to sum values of an array.**

**A.** public class sumOfArrayElements{

public static void main(String[] args){

int array[]={1,2,3,4,5,6,7,8,9,10,11};

int sum=0;

for(int i:array)

sum=sum+I;

System.out.priintln(“The sum of array elements is ”+ sum);

}

}

**24.Write a Java Program to sort numeric array and string array.**

**A.**

Import java.util.Arrays;

Public class SortArray{

Public static void main(String[] args){

Int arr1[]={25,65,98,22,3,9,45};

String arr2[]={“sowmya”,”praneeeth”,”nikitha”,”Vishnu”,”jayanth”};

System.out.println(“Original array:”+Arrays.toString(arr1));

Arrays.sort(arr1);

System.out.println(“Sorted numerical array:”+Arrays.toString(arr1));

System.out.println(“Original String array:”+Arrays.toString(arr2));

Arrays.sort(arr2);

System.out.println(“Sorted array:”+Arrays.toString(arr2));

}

}