1. Can abstract class have constructors in Java?

Ans) Yes when we define a class to be an Abstract Class it cannot be instantiated but that does not mean an Abstract class cannot have a constructor. Each abstract class must have a concrete subclass which will implement the abstract methods of that abstract class.

1. Can abstract class implements interface in Java? do they require to implement all methods?

Ans) In Java, an abstract class can implement an interface, and not provide implementations of all of the interface's methods. It is the responsibility of the first concrete class that has that abstract class as an ancestor to implement all of the methods in the interface. The only way to prevent this is if the subclass is also declared abstract , so that it cannot be instantiated in the first place. You don't have to implement all methods of an abstract class. But you must implement all abstract methods of it.

3)  Can abstract class be final in Java?

Ans) Yes, there may be "final" methods in "abstract" class. But, any "abstract" method in the class can't be declared final.In Abstract Class methods may be defined or not. If we extend the abstract class then only it has meaning, so what ever methods we declare or defined in Abstract call it will over ride in subclass.  
4)  Can abstract class have static methods in Java?

Ans) you can't declare a static method to be abstract. Normally, the compiler can guarantee that an abstract method will have a real implementation any time that it is called, because you can't create an instance of anabstract class.  
5)  Can you create instance of abstract class?

Ans) No, **you** cannot **create** an **instance** of an **abstract class** because it **does** not have a complete implementation. The purpose of an **abstract class** is to function as a base for subclasses. It acts like a template, or an empty or partially empty structure,**you** should extend it and build on it before **you can** use it.  
6)  Is it necessary for abstract class to have abstract method?

Ans) Yes we can **have** an **abstract class** without **Abstract Methods** as both are independent concepts. Declaring a **class abstract** means that it can not be instantiated on its own and can only be sub classed. Declaring a **method abstract**means that **Method** will be defined in the subclass.  
7)  Difference between abstract class and interface in Java?

Ans) Main **difference** is methods of a **Java interface** are implicitly **abstract** and cannot have implementations. A **Java abstract class** can have instance methods that implements a default behavior. 2.Variables declared in a **Java interface** is by default final. An**abstract class** may contain non-final variables.  
8)  When do you favor abstract class over interface?

Ans) Abstract classes allow you to provide default functionality for the subclasses. Common knowledge at this point. Why is this extremely important though? If you plan on updating this base class throughout the life of your program, it is best to allow that base class to be an abstract class. Why? Because you can make a change to it and all of the inheriting classes will now have this new functionality

9)    What is abstract method in Java?

Ans) **Abstract** classes cannot be instantiated, but they can be subclassed. An **abstract method** is a **method** that is declared without an implementation (without braces, and followed by a semicolon),

like this:

**abstract** void sum(double X, double Y);

10) Can abstract class contains main method in Java ?

Ans) So there's no problem. **Abstract** just means you **can**'t instantiate the **class** directly. You **can have** constructors if you want - they might be needed for subclasses to initiate the object state. You **can have** static **methods**, including **main**() and they don't need an object so calling them is fine.

11)  what is static block in java?

Ans) **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?

Ans) **Static Block**. **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.

13)  Can we overload static methods in java?

Ans) **Static methods** cannot be overridden because they are not dispatched on the object instance at runtime. The compiler decides which **method** gets called. ... No,**Static methods can**'t be overriden as it is part of a class rather than an object. But one **can overload static method**.

14)  Can we call super class static methods from sub class?

Ans) You **can** actually **call** the **static method** of a **superclass** in a generic way, given that you know the **method** name and its parameters. This should work and in the**subclass** you have everything set in the base **class** available. For **static methods**there is no instance of a **class** needed, so there is no **super**

15)What is the difference between final and static keywords?

Ans) In the case of **final** variables, they should either be assigned at declaration or in the constructor. In the case of **final** classes, it means that they cannot be subclassed.**Static**: ... So if a class has a **static** variable, no matter how many instances of it you create, they all would have the same value for the variable.

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

Ans) The covariant return type specifies that the return type may vary in the same direction as the subclass.

Before Java5, it was not possible to override any method by changing the return type. But now, since Java5, it is possible to override method by changing the return type if subclass overrides any method whose return type is Non-Primitive but it changes its return type to subclass type

For example:-

class A{

A get(){return this;}

}

class B1 extends A{

B1 get(){return this;}

void message(){System.out.println("welcome to covariant return type");}

public static void main(String args[]){

new B1().get().message();

}

}

17) Write a note on Enum with example code.

Ans) **Enum in java** is a data type that contains fixed set of constants.

It can be used for days of the week (SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY and SATURDAY) , directions (NORTH, SOUTH, EAST and WEST) etc. The java enum constants are static and final implicitly. It is available from JDK 1.5.

class EnumExample1{

public enum Season { WINTER, SPRING, SUMMER, FALL }

public static void main(String[] args) {

for (Season s : Season.values())

System.out.println(s);

}}

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

Ans) The **super** keyword in java is a reference variable which is used to refer immediate parent class object.Whenever you create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.

It is used inside a sub-class method definition to call a method defined in the superclass. Private methods of the super-class cannot be called. Only public and protected methods can be called by the super keyword. It is also used by class constructors to invoke constructors of its parent class.

19)  Write a code to implement abstraction using interface.

Ans) interface A{

void a();

void b();

void c();

void d();

}

abstract class B implements A{

public void c(){System.out.println("I am C");}

}

class M extends B{

public void a(){System.out.println("I am a");}

public void b(){System.out.println("I am b");}

public void d(){System.out.println("I am d");}

}

class Test5{

public static void main(String args[]){

A a=new M();

a.a();

a.b();

a.c();

a.d();

}}

20)Write a Java program to sort a numeric array and a string array.

Ans)

import java.io.\*;

import java.util.Arrays;

class sort

{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

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

{

System.out.println("1.Sort String array");

System.out.println("2.Sort int array");

System.out.println("Enter your choice");

int ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1: sortintArray();

break;

case 2: sortStringArray();

break;

}

}

void sortintArray()

{

int arr[]=new int[10];

System.out.println("Please enter:-"+arr.length+" elements");

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

{

arr[i]=Integer.parseInt(br.readLine());

}

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

{

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

{

if(arr[j]<arr[i])

{

int temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

}

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

{

System.out.println(arr[i]);

}

}

void sortStringArray()

{

String srr[]=new String[10];

System.out.println("Please enter:-"+srr.length+" elements");

for(int i=0;i<srr.length;i++)

{

srr[i]=br.readLine();

}

Arrays.sort(srr);

for(int i=0;i<srr.length;i++)

{

System.out.println(srr[i]);

}

}

}

21)Write a Java program to sum values of an array.

Ans) import java.io.\*;

class sum

{

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

{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int arr[]=new int[10];

int sum=0;

System.out.println("Please enter:-"+arr.length+" elements");

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

{

arr[i]=Integer.parseInt(br.readLine());

s=s+arr[i];

}

System.out.println("Sum of Array all elements="+s);

}

}

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

Ans)

import java.util.Arrays;

public class remove {

public static void main(String[] args) {

int arr[] = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};

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 second element: "+Arrays.toString(arr));

}

}

23)Write a Java program to reverse an array of integer values.

Ans)

import java.util.Arrays;

public class reverse {

public static void main(String[] args){

int[] arr1 = {

1789, 2035, 1899, 1456, 2013,

1458, 2458, 1254, 1472, 2365,

1456, 2165, 1457, 2456};

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

for(int i = 0; i < arr1.length / 2; i++)

{

int temp = arr1[i];

arr1[i] = arr1[arr1.length - i - 1];

arr1[arr1.length - i - 1] = temp;

}

System.out.println("Reverse array : "+Arrays.toString(arr1));

}

}

24)Write a Java program to find the duplicate values of an array of integer values.

Ans) import java.util.Arrays;

public class Exercise12 {

public static void main(String[] args)

{

int[] arr = {1, 2, 5, 5, 6, 6, 7, 2};

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]);

}

}

}

}

}