1. Can abstract class have constructors in Java?

Ans: Yes, an abstract class can have a constructor in the following scenarios:

* When we want to initialise the data members of the abstract class before the instantiation of a subclass.
* In case we have not initialised the final fields in an abstract class during the declaration then constructors are required.

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

Ans: Yes, an abstract class can implement interface in Java.

No they don’t require implementing all the methods of the interface as by definition the child class of the abstract needs to implement the methods of interface that is implemented by the abstract class.

3)  Can abstract class be final in Java?

Ans: No a class cannot be marked "abstract" as well as "final” .Defining a class "abstract" means it contains partial implementation of methods and hence depends on its subclasses to provide concrete implementation. If we declare a class "final" then no other class can extend it.

4)  Can abstract class have static methods in Java?

Ans: In Java you can have a static method in an abstract class.

This is allowed because that method can be called directly, without creating instance of the abstract class. But we can’t declare a static method to be abstract. Since a static method can be called directly, making it abstract would lead calling an undefined function.

5)  Can you create instance of abstract class?

Ans: No, we cannot create an instance of an abstract class because it does not have a complete implementation. It serves as a base or a template for subclasses. Hence we need to extend it.

6)  Is it necessary for abstract class to have abstract method?

Ans: No, it is not necessary for abstract class to have abstract methods Declaring a class abstract means that it cannot be instantiated on its own and can only be inherited and used by its sub class. Declaring a method abstract means that method will be defined in the subclass.  
7)  Difference between abstract class and interface in Java?

Ans: An interface is an empty structure. There are only the signatures of the methods, which imply that the methods do not have a body. It is a pattern to be used by the classes implementing it. Implementing an interface consumes very little CPU and there isn't any expensive look-up to do.

Abstract classes, unlike interfaces, are classes. They are more expensive to use, because there is a look-up to do when we inherit from them. Abstract classes can have a behaviour defined for them.

8)  When do you favor abstract class over interface?

Abstract classes allow for default default function definition. If we have a base class where all the classes will perform the same function, then we can define that in our Abstract class.

If we are creating something for objects that are closely related in a hierarchy, then abstract class is preferred. An example of this would be something like a business rules engine. This engine would take in multiple Business Rules as classes and each one of these classes will have an analysis function on it.

9)    What is abstract method in Java?

Ans: A method without body i.e. no implementation is known as abstract method. An abstract method must always be declared in an abstract class.

Example: public abstract int myMethod(int n1, int n2);

10) Can abstract class contains main method in Java ?

Ans: Yes, We can have main() method in abstract class. Loading a class is not the same as creating an instance of the class. And there's no need to create an instance of the class to call main(), because it's static

Example:

public abstract class Sample

{

public static void main(String args[])

{

System.out.println("Abstract Class main method : ");

}

}

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?

If a class has static members that require complex initialization, a static block is the tool to use.

public static final Map<String, String> map = new HashMap<String, String>();

static

{

map.put("AEN", "Alfred E. Newman");

}

13)  Can we overload static methods in java?

|  |
| --- |
| Ans: Yes,we can overload static methods. We can have two or more static methods with same name, but differences in input parameters.  Example:  public class Test {      public static void func() {          System.out.println("Test.func() called ");      }      public static void func(int a) {          System.out.println("Test.func(int) called ");      }      public static void main(String args[])      {          Test.func();          Test.func(10);      }  } |
|  |
|  |

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

Ans:Yes we can call super class static method inside sub class using super\_class\_name.method();

We can also call super class static method using Sub\_class\_name.superclass\_staticMethod();

If the same static method defined in sub class also then we can not call super class method using sub class name if we call them sub class static method will be executed.

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

Ans: ‘final’ means the values once assigned cannot be modified. In the case of final variables, they should either be assigned at declaration or in the constructor.

Static basically means that the values would be stored in the class memory. 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.

Example:

class SuperClass{

SuperClass  get(){return this;}

}

class SubClass extends A{

SubClass get(){return this;}

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

public static void main(String args[]){

new SubClass().get().message();

}

}

17) Write a note on Enum with example code.

Ans: An enum type is a special data type that enables for a variable to be a set of predefined constants. The variable must be equal to one of the values that have been predefined for it.

Example:

class EnumExample{

public enum Season { WINTER, SPRING, SUMMER }

  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.

super() calls the parent constructor with no arguments. It can be used also with arguments by using the syntax : super(argument1,… argument n)

19)  Write a code to implement abstraction using interface.

interface MyInterface

{

public void method1();

public void method2();

}

class Demo implements MyInterface

{

// This class must have to implement both the abstract methods

public void method1()

{

System.out.println("implementation of method1");

}

public void method2()

{

System.out.println("implementation of method2");

}

public static void main(String arg[])

{

MyInterface obj = new Demo();

obj.method1();

}

}

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

import java.util.Arrays;

public class sorting {

public static void main(String[] args){

int[] int\_array = {20,30,11,5,33,45,2};

String[] string\_array = {“Hello”, ”World”, ”Java”, ”Task” };

System.out.println("Original numeric array : "+Arrays.toString(int\_array));

Arrays.sort(int\_array);

System.out.println("Sorted numeric array : "+Arrays.toString(int\_array));

System.out.println("Original string array : "+Arrays.toString(string\_array));

Arrays.sort(string\_array);

System.out.println("Sorted string array : "+Arrays.toString(string\_array));

}

}

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

import java.util.Scanner;

class SumDemo{

public static void main(String args[]){

Scanner scanner = new Scanner(System.in);

int sum = 0;

System.out.println("Enter the total no. of elements:");

int n=scanner.nextInt();

int[] array = new int[n];

System.out.println("Enter the elements:");

for (int i=0; i<n; i++)

{

array[i] = scanner.nextInt();

}

for( int num : array)

{

sum = sum+num;

}

System.out.println("Sum of array elements is:"+sum);

}

}

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

import java.util.Arrays;

public class ArayRemoveElement {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int removeIndex ;

System.out.println("Enter the total no. of elements:");

int n=scanner.nextInt();

int[] array = new int[n];

System.out.println("Enter the elements:");

for (int i=0; i<n; i++)

{

array[i] = scanner.nextInt();

}

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

System.out.println("Enter the index of element you wish to remove”);

removeIndex=scanner.nextInt();

if(removeIndex>0 && removeIndex<n)

{

for(int i = removeIndex; i < array.length -1; i++){

array[i] = array[i + 1];

}

}

System.out.println("After removing the “+(removeIndex+1)+ “ element: "+Arrays.toString(array));

}

}

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

public class ReverseArray

{

     public static void main(String[] args) {

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

       System.out.println("Array before reverse:");

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

{

            System.out.print(numbers[i] + " ");

        }

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

             int temp = numbers[i];

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

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

        }

        System.out.println("\nArray after reverse:");

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

            System.out.print(numbers[i] + " ");

        }

    }

}

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

import java.util.Arrays;

public class ArrayDuplicateElements{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

int removeIndex ;

System.out.println("Enter the total no. of elements:");

int n=scanner.nextInt();

int[] array = new int[n];

System.out.println("Enter the elements:");

for (int i=0; i<n; i++)

{

array[i] = scanner.nextInt();

}

for (int i = 0; i < array.length-1; i++)

{

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

{

if ((array[i] == array[j]) && (i != j))

{

System.out.println("Duplicate Element : "+array[j]);

}

}

}

}

}