Q1(a): How is java different from c++?Explain the working of JRE.

Answer: C++ is compiled and run using the compiler which converts source code into machine code so, C++ is platform dependent. Java uses both compiler and interpreter. Java source code is converted into bytecode at compilation time. The interpreter executes this bytecode at runtime and produces output.

The JRE is the underlying technology that communicates between the Java program and the operating system. It acts as a translator and facilitator, providing all the resources so that once you write Java software, it runs on any operating system without further modifications.

Q1(b):What is role of methods in java? How can we use ‘this’ keyword with local variables in java.

Answer: The primary uses of methods in Java are: It allows code reusability (define once and use multiple times) You can break a complex program into smaller chunks of code. It increases code readability. Class Method. A Method provides information about, and access to, a single method on a class or interface. The reflected method may be a class method or an instance method (including an abstract method).

In Java, this is a reference variable that refers to the current object on which the method or constructor is being invoked. It can be used to access instance variables and methods of the current object.

**Below is the implementation of this reference:**

|  |
| --- |
| // Java Program to implement  // Java this reference    // Driver Class  public class Person {        // Fields Declared      String name;      int age;        // Constructor      Person(String name, int age)      {          this.name = name;          this.age = age;      }        // Getter for name      public String get\_name() { return name; }        // Setter for name      public void change\_name(String name)      {          this.name = name;      }        // Method to Print the Details of      // the person      public void printDetails()      {          System.out.println("Name: " + name);          System.out.println("Age: " + age);          System.out.println();      }        // main function      public static void main(String[] args)      {          // Objects Declared          Person first = new Person("ABC", 18);          Person second = new Person("XYZ", 22);            first.printDetails();          second.printDetails();            first.change\_name("PQR");          System.out.println("Name has been changed to: "                             + first.get\_name());      }  } |

**Output**

Name: ABC

Age: 18

Name: XYZ

Age: 22

Name has been changed to: PQR

Q2(a):What is the use of string class In java?Explain any of its three methods.

Answer: Java String class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc. The java.lang.String class implements Serializable, Comparable and CharSequence interfaces.

1indexOf():

Used to find characters and substrings in a string. It returns the index of the first occurrence from the left of the passed string within this string; moreover, if we provide the fromIndex, it starts searching from that index.

public static int findOccurances(String[] paragraph, String target) {

     int index = 0, totalOccurances = 0;

 for (int lineNumber = 0;lineNumber < paragraph.length; ++lineNumber)

 {

   String line = paragraph[lineNumber];

   while (true) {

    index = line.indexOf(target, index);

    if (index != -1) {

  System.out.println("Found at line number " + lineNumber + " at position " + index);

    totalOccurances++;

    index += target.length();

  } else {

    break;

   }

  }

 }

 return totalOccurances;

}

2. toCharArray():Used to form a new character array from this string. The newly allocated array's contents are initialized to contain the characters represented by this string, and its length is the length of this string.

String palindrome = "Dot saw I was Tod";

int len = palindrome.length();

char[] charArray = palindrome.toCharArray();

for (int j = 0; j < len; j++) {

    if (charArray[j] != charArray[len - 1 - j]) {

       System.out.println(“Not a palindrome”);

    }

}

System.out.println(“Valid palindrome”);

3. concat():The passed string is appended to the end of the specified string. It's a great option for combining/merging strings.

String addressLine1 = "4424  Black Oak Hollow Road ";

  String addressLine2 = "San Francisco ";

  String addressLine3 = "California 94108";

  String fullAddress =

  addressLine1.concat(addressLine2).concat(addressLine3);

  System.out.println(fullAddress);

Q2(b):Why are exceptions in java?How can we handle these?

Answer: **In Java, Exception** is an unwanted or unexpected event, which occurs during the execution of a program, i.e. at run time, that disrupts the normal flow of the program’s instructions. Exceptions can be caught and handled by the program. When an exception occurs within a method, it creates an object. This object is called the exception object. It contains information about the exception, such as the name and description of the exception and the state of the program when the exception occurred.

**Major reasons why an exception Occurs**

* Invalid user input
* Device failure
* Loss of network connection
* Physical limitations (out-of-disk memory)
* Code errors
* Opening an unavailable file

**Errors** represent irrecoverable conditions such as Java virtual machine (JVM) running out of memory, memory leaks, stack overflow errors, library incompatibility, infinite recursion, etc. Errors are usually beyond the control of the programmer, and we should not try to handle errors.

1. public class JavaExceptionExample{
2. public static void main(String args[]){
3. try{
4. //code that may raise exception
5. int data=100/0;
6. }catch(ArithmeticException e){System.out.println(e);}
7. //rest code of the program
8. System.out.println("rest of the code...");
9. }
10. }

Q3(a):What do you understand by constructor overloading? Explain with example how it is different from method overriding?

Answer: The process of defining multiple constructors of the same class is referred to as Constructor overloading. However, each constructor should have a different signature or input parameters. In other words, constructor overloading in Java is a technique that enables a single class to have more than one constructor that varies by the list of arguments passed. Each overloaded constructor is used to perform different task in the class.

The Java compiler identifies the overloaded constructors on the basis of their parameter lists, parameter types and the number of input parameters. Hence, the constructors that are overloaded should have different signatures. A constructor’s signature contains its name and parameter types. An ambiguity issue arises when two of the class constructors have an identical signature.

The compiler fails to differentiate between the two and hence returns an error notification. When constructors with different signatures are overloaded, the compiler determines the constructor to be invoked based on the number of input parameters of the objects.

// An example class to understand need of

// constructor overloading.

class Box

{

double width, height,depth;

// constructor used when all dimensions

// specified

Box(double w, double h, double d)

{

width = w;

height = h;

depth = d;

}

// compute and return volume

double volume()

{

return width \* height \* depth;

}

}

Overriding occurs when the method signature is the same in the superclass and the child class. Overloading occurs when two or more methods in the same class have the same name but different parameters.

Q3(b):How can we use final keyword in java?What are arrays of objects?

Answer: Set a variable to final, to prevent it from being overridden/modified:

public class Main {

**final** int x = 10;

public static void main(String[] args) {

Main myObj = new Main();

myObj.x = 25; // will generate an error: cannot assign a value to a **final** variable

System.out.println(myObj.x);

}

}

Java is an object-oriented programming language. Most of the work done with the help of objects. We know that an array is a collection of the same data type that dynamically creates objects and can have elements of primitive types. Java allows us to store objects in an array. In Java, the class is also a user-defined data type. An array that conations class type elements are known as an array of objects. It stores the reference variable of the object.

// Java program to demonstrate an array

// of objects is declared with initial values.

class GFG {

    public static void main(String args[])

    {

        // Creating an array of objects

        // declared with initial values

        Object[] javaObjectArray

            = { "Maruti", new Integer(2019), "Suzuki",

                new Integer(2019) };

      // Printing the values

        System.out.println(javaObjectArray[0]);

        System.out.println(javaObjectArray[1]);

        System.out.println(javaObjectArray[2]);

        System.out.println(javaObjectArray[3]);

    }

}

Q4(a):What are 2d arrays in java?Write a program to print sum of all positive integer array values.

Answer: **Multidimensional Arrays** can be defined in simple words as array of arrays. Data in multidimensional arrays are stored in tabular form (in row major order).

**Syntax:**

**data\_type**[1st dimension][2nd dimension][]..[Nth dimension] **array\_name** = **new data\_type**[size1][size2]….[sizeN];

**where:**

* **data\_type**: Type of data to be stored in the array. For example: int, char, etc.
* **dimension**: The dimension of the array created. For example: 1D, 2D, etc.
* **array\_name**: Name of the array
* **size1, size2, …, sizeN**: Sizes of the dimensions respectively.

**Example:**

Two dimensional array:  
int[][] twoD\_arr = new int[10][20];

public class SumOfArray {

1. public static void main(String[] args) {
2. //Initialize array
3. int [] arr = new int [] {1, 2, 3, 4, 5};
4. int sum = 0;
5. //Loop through the array to calculate sum of elements
6. for (int i = 0; i < arr.length; i++) {
7. sum = sum + arr[i];
8. }
9. System.out.println("Sum of all the elements of an array: " + sum);
10. }
11. }

Q4(b):write down the difference b/w instance variable and class variable with suitable example.write a program to show the working of static and non static blocks in java.

Answer:

|  |  |
| --- | --- |
| **Instance variables** | **Static (class) variables** |
| [Instance variables](https://www.tutorialspoint.com/Instance-variables-in-Java) are declared in a class, but outside a method, constructor or any block. | Class variables also known as [static variables](https://www.tutorialspoint.com/class-and-static-variables-in-java) are declared with the static keyword in a class, but outside a method, constructor or a block. |
| Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed. | Static variables are created when the program starts and destroyed when the program stops. |
| Instance variables can be accessed directly by calling the variable name inside the class. However, within static methods (when instance variables are given accessibility), they should be called using the fully qualified name. *ObjectReference.VariableName*. | Static variables can be accessed by calling with the class name *ClassName.VariableName*. |
| Instance variables hold values that must be referenced by more than one method, constructor or block, or essential parts of an object's state that must be present throughout the class. | There would only be one copy of each class variable per class, regardless of how many objects are created from it. |

**Example**

public class VariableExample{

int myVariable;

static int data = 30;

public static void main(String args[]){

VariableExample obj = new VariableExample();

System.out.println("Value of instance variable: "+obj.myVariable);

System.out.println("Value of static variable: "+VariableExample.data);

}

}

1. public class StaticBlock
2. {
4. // Constructor of the class StaticBlock
5. StaticBlock()
6. {
7. System.out.println("Inside the constructor of the class.");
8. }
10. // print method of the StaticBlock class
11. public static void print()
12. {
13. System.out.println("Inside the print method.");
14. }
16. static
17. {
18. System.out.println("Inside the static block.");
19. }
21. // main method
22. public static void main(String[] args)
23. {
25. // instantiating the class StaticBlock
26. StaticBlock sbObj = new StaticBlock();
27. sbObj.print(); // invoking the print() method
29. // invoking the constructor inside the main() method
30. new StaticBlock();
32. }
33. }

class Student {

Student(){

System.out.println("Constructor");

}

// non-static block

{

System.out.println("Non-static block");

}

public static void main(String[] args) {

Student student = new Student();

}

}

Q5(a):What are command line arguments in java?Write a program to show the use of wrapper class for printing sum of all positive integers as passed through command line arguments.

Answer: Java command-line argument is an argument i.e. passed at the time of running the Java program. In Java, the command line arguments passed from the console can be received in the Java program and they can be used as input. The users can pass the arguments during the execution bypassing the command-line arguments inside the main() method.

// Java Program to Illustrate First Argument

// Class

class GFG {

    // Main driver method

    public static void main(String[] args)

    {

        // Printing the first argument

        System.out.println(args[0]);

    }

}

Sum of integers:

class SumTest {

public static void main(String[] values) {

int sum = 0;

System.out.println("Calculates Sum for below Integers");

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

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

sum = sum + Integer.parseInt(values[i]);

}

System.out.println("Sum :" + sum);

}

}

Q5(b):What is the use of char data type in java? Write a program to override any of object class method.

Ansswer: The char keyword is a data type that is used to store a single character. A char value must be surrounded by single quotes, like 'A' or 'c'.

char myGrade = 'B';

System.out.println(myGrade);

 //Java Program to illustrate the use of Java Method Overriding

 //Creating a parent class.

 class Vehicle{

   //defining a method

   void run(){System.out.println("Vehicle is running");}

 }

 //Creating a child class

 class Bike2 extends Vehicle{

   //defining the same method as in the parent class

   void run(){System.out.println("Bike is running safely");}



   public static void main(String args[]){

   Bike2 obj = new Bike2();//creating object

   obj.run();//calling method

   }

 }