**Java – Multithreading – Programs**

Q1: In the following java program, what is the name of the thread?

class multithreaded\_programing

{

public static void main(String args[])

{

Thread t = Thread.currentThread();

System.out.println(t);

}

}

a. Thread

b. main

c. System

d. t

Q2: Name the method of the thread that is called before the run() method and carries out initialization.

a. resume()

b. start()

c. suspend()

d. begin()

Q3: Predict the output of the following code:

class MyThread extends Thread

{

public void run()

{

System.out.println("Running");

}

}

class ThreadTest {

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

{

Runnable r = new MyThread(); // #1

Thread myThread = new Thread(r); // #2

myThread.start();

}

}

a. Error on #1

b. Error on #2

c. Running

d. No output

Q4: O/p of following program

public class ThreadName extends Thread {

public void run()

{

for(int i = 0; i < 3; i++)

{

System.out.println("C");

System.out.println("D");

}

}

public static void main(String args[])

{

ThreadTest t1 = new ThreadTest();

ThreadName t2 = new ThreadName();

t1.start();

t2.start();

}

}

class ThreadTest extends Thread

{

public void run()

{

for(int i = 0; i < 3; i++)

{

System.out.println("A");

System.out.println("B");

}

}

}

a. Compile Error

b. Run time error

c. Prints A B C D, but order cannot be guaranteed

d. Prints ABCD in order

Answer: C

Q5: Which method registers a thread in a thread scheduler?

a. run();

b. start();

c. register();

d. begin();

Answer: b

Q6: O/p of following

class ThreadDemo extends Thread

{

public static void main(String [] args)

{

ThreadDemo t = new ThreadDemo();

t.start();

System.out.print("one. ");

t.start();

System.out.print("two. ");

}

public void run()

{

System.out.print("Thread ");

}

}

a. Compile Error

b. It prints "Thread one. Thread two.".

c. An exception occurs at runtime.

d. The output cannot be determined.

Answer: c

Q7: O/p of following:

public class ThreadName extends Thread {

public static void main(String [] args)

{

ThreadName t = new ThreadName(); /\* Line 5 \*/

t.run(); /\* Line 6 \*/

}

public void run()

{

for(int i=1; i < 3; ++i)

{

System.out.print(i + "..");

}

}

}

a. Compile error

b. Run time error

c. 1..2..

Answer: c

Q8: Which method must be defined by a class implementing the java.lang.Runnable interface?

a. public void start()

b. public void run()

c. void run()

d. void start()

Answer: b

Q9: Which of the following piece of code will create and start this thread?

public class MyRunnable implements Runnable

{

public void run()

{

// some code here

}

}

a. new Thread(MyRunnable).run();

b. new Runnable(MyRunnable).start();

c. new Thread(new MyRunnable()).start();

d. new MyRunnable().start();

Answer: c

**Java – Multithreading - Theory**

Q1: What is a thread?

A:

A thread is an extremely lightweight process, or the smallest component of the process, that enables software to work more effectively by doing numerous tasks concurrently.

Q2: What is the life cycle of a thread?

A:

New -> Runnable -> Running -> Waiting -> Dead

Q3: What is multithreading?

A: Multithreading in Java is a process of executing multiple threads simultaneously. The main reason for incorporating threads into an application is to improve its performance. Games and animations can also be made using threads.

Q4: What are the two ways of implementing thread in Java? Show with an example

A:

First Way: Extending the Thread class

class MultithreadingDemo extends Thread

{

public void run()

{

System.out.println("My thread is in running state.");

}

public static void main(String args[])

{

MultithreadingDemo obj=new MultithreadingDemo();

obj.start();

}

}

Second Way: Implementing Runnable interface in Java

class MultithreadingDemo implements Runnable

{

public void run()

{

System.out.println("My thread is in running state.");

}

public static void main(String args[])

{

MultithreadingDemo obj=new MultithreadingDemo();

Threadtobj =new Thread(obj); tobj.start();

}

}

Q5: What is the start() and run() method of Thread class?

A:

start(): In simple words, the start() method is used to start or begin the execution of a newly created thread. When the start() method is called, a new thread is created and this newly created thread executes the task that is kept in the run() method. One can call the start() method only once.

run(): In simple words, the run() method is used to start or begin the execution of the same thread. When the run() method is called, no new thread is created as in the case of the start() method. This method is executed by the current thread. One can call the run() method multiple times.

Q6: Explain thread pool?

A:

A Thread pool is simply a collection of pre-initialized or worker threads at the start-up that can be used to execute tasks and put back in the pool when completed. It is referred to as pool threads in which a group of fixed-size threads is created. By reducing the number of application threads and managing their lifecycle, one can mitigate the issue of performance using a thread pool. Using threads, performance can be enhanced and better system stability can occur. To create the thread pools, java.util.concurrent.Executors class usually provides factory methods.

Q7: Can you start a thread twice?

A:

No, it's not at all possible to restart a thread once a thread gets started and completes its execution. Thread only runs once and if you try to run it for a second time, then it will throw a runtime exception i.e., java.lang.IllegalThreadStateException.

**Java – Collections – Programs**

Q1: Which interface provides key-value pair?

a. List

b. Set

c. Map

d. Collection

Answer: c

Q2: O/p of following program:

import java.util.\*;

class UtilitiesTest {

public static void main(String[] args) {

List < int > intList = new ArrayList < > ();

intList.add(10);

intList.add(20);

System.out.println("The list is: " + intList);

}

}

A. It prints the following: The list is: [10, 20]

B. It prints the following: The list is: [20, 10]

C. It results in a compiler error

D. It results in a runtime exception

Answer: C

Q3: Which of these maintains insertion order?

a.List

b.Set

c.All

d.None

Answer: a

Q4: Which data structure ArrayList internally uses?

a. LinkedList

b. Array

c. Doubly LinkedList

d. None

Answer: b

Q5: What is the default value of load factor in a hashing based collection?

a. 0.75

b. 0.78

c. 0.6

d. 0.5

Answer: a

**Java – Collections – Theory**

Q1) What is the Collection framework in Java?

A: Collection Framework is a combination of classes and interface, which is used to store and manipulate the data in the form of objects. It provides various classes such as ArrayList, Vector, Stack, and HashSet, etc. and interfaces such as List, Queue, Set, etc. for this purpose.

Q2)What are the main differences between array and collection?

A: Array and Collection are somewhat similar regarding storing the references of objects and manipulating the data, but they differ in many ways. The main differences between the array and Collection are defined below:

a. Arrays are always of fixed size, i.e., a user can not increase or decrease the length of the array according to their requirement or at runtime, but In Collection, size can be changed dynamically as per need.

b. Arrays can only store homogeneous or similar type objects, but in Collection, heterogeneous objects can be stored.

c. Arrays cannot provide the ?ready-made? methods for user requirements as sorting, searching, etc. but Collection includes readymade methods to use.

Q3) What is the List interface ?

A: List interface extends the Collection interface, and it is an ordered collection of objects. It contains duplicate elements. It also allows random access of elements.

Syntax:

public interface List<E> extends Collection<E>

Q4) What is the Set interface ?

A: Set (java.util.Set) interface is a collection which cannot contain duplicate elements. It can only include inherited methods of Collection interface

Syntax:

public interface Set<E> extends Collection<E>

Q5)What is the Queue interface ?

A: Queue (java.util.Queue) interface defines queue data structure, which stores the elements in the form FIFO (first in first out).

Syntax:

public interface Queue<E> extends Collection<E>

Q6)What is the Map interface ?

A: A Map (java.util.Map) represents a key, value pair storage of elements. Map interface does not implement the Collection interface. It can only contain a unique key but can have duplicate elements. There are two interfaces which implement Map in java that are Map interface and Sorted Map.

**Java – Exceptions,Strings, Utility Methods – Programs**

Q1: O/p of following program

class Test extends Exception { }

class Main {

public static void main(String args[]) {

try {

throw new Test();

}

catch(Test t) {

System.out.println("Got the Test Exception");

}

finally {

System.out.println("Inside finally block ");

}

}

}

A: Got the Test Exception

Inside finally block

B: Got the Test Exception

C: Inside finally block

D: Compiler Error

Q2: O/p of following program

class Main {

public static void main(String args[]) {

int x = 0;

int y = 10;

int z = y/x;

}

}

A: Compiler Error

B: Compiles and runs fine

C: Compiles fine but throws ArithmeticException exception

Q3: O/p of following program

class Base extends Exception {}

class Derived extends Base {}

public class Main {

public static void main(String args[]) {

// some other stuff

try {

// Some monitored code

throw new Derived();

}

catch(Base b) {

System.out.println("Caught base class exception");

}

catch(Derived d) {

System.out.println("Caught derived class exception");

}

}

}

A: Caught base class exception

B: Caught derived class exception

C: Compiler Error because derived is not throwable

D: Compiler Error because base class exception is caught before derived class

Q4: O/p of following program

class Test

{ int count = 0;

void A() throws Exception

{

try

{

count++;

try

{

count++;

try

{

count++;

throw new Exception();

}

catch(Exception ex)

{

count++;

throw new Exception();

}

}

catch(Exception ex)

{

count++;

}

}

catch(Exception ex)

{

count++;

}

}

void display()

{

System.out.println(count);

}

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

{

Test obj = new Test();

obj.A();

obj.display();

}

}

A: 4

B: 5

C: 6

D: Compilation error

Q5: O/p of the following program?

public class JavaExceptionHandlingQuiz

{

public static void main(String[] args)

{

try

{

int[] a = {1, 2, 3, 4};

int i = a[4];

}

catch (NumberFormatException e)

{

System.out.println(1);

}

catch (NullPointerException e)

{

System.out.println(2);

}

catch (ArrayIndexOutOfBoundsException e)

{

System.out.println(3);

}

}

}

A: Compilation Error

B: 3

C: 2

D: 1

Q6:O/p of the program:

public class StrEqual {

public static void main(String[] args) {

String s1 = "hello";

String s2 = new String("hello");

String s3 = "hello";

if (s1 == s2) {

System.out.println("s1 and s2 equal");

} else {

System.out.println("s1 and s2 not equal");

}

if (s1 == s3) {

System.out.println("s1 and s3 equal");

} else {

System.out.println("s1 and s3 not equal");

}

}

}

A:

Q7: O/p of following program

public class Test {

public static void main(String[] args) {

String s = new String("5");

System.out.println(1 + 10 + s + 1 + 10);

}

}

Q8: O/p of the program

public class A {

public static void main(String[] args) {

String s1 = new String("Monday");

String s2 = new String("tuesday");

System.out.println(s1 = s2);

}

}

Q9: How many objects will be created in the below program and why?

String s1 = "Java";

String s2 = "Java";

A: 1

Q10: How many objects will be created in the below program and why?

String s1 = new String("Java");

String s2 = "Java";

Q11: O/p of below program?

String s = "Today is Monday";

System.out.println(s.charAt(s.toUpperCase().length()));

A. Runtime Exception

B. Prints “y”

C. Prints “Y”

D. Compilation Error

Q: O/p of following?

public class Test {

public static void main(String[] args) {

String x = "abc";

String y = "abc";

x.concat(y);

System.out.print(x);

}

}

A. abc

B. abcabc

C. null

Q12: O/p of following:

public class Test {

public static void main(String[] args) {

String s1 = "abc";

String s2 = "abc";

System.out.println("s1 == s2 is:" + s1 == s2);

}

}

A. false

B. s1 == s2 is:true

C. s1 == s2 is:false

D. true

Q13: What is wrong with below code?

class MyStringClass extends String

{

String string;

}

Q14: O/p of following:

System.out.println("Java"+1000+2000+3000);

Q15: O/p of following:

System.out.println(1000+2000+3000+"Java");

Q16: What is the default capacity of StringBuffer?

A:

Q17: O/p of following code

A:

public class JavaStringsQuiz

{

public static void main(String[] args)

{

StringBuilder sb = new StringBuilder(-32);

sb.append("ABC");

System.out.println(sb);

}

}

**Java – Exceptions,Strings, Utility Methods – Theory**

Q1: How can you handle exceptions in Java?

A:

Exception handling can be performed using:

Try: the set of statements or code which requires monitoring for an exception is kept under this block.

Catch: this block catches all exceptions that were trapped in the try block.

Finally: this block is always performed irrespective of the catching of exceptions in the try or catch block.

Q2: What is the difference between exception and error in Java?

A: Errors typically happen while an application is running. For instance, Out of Memory Error occurs in case the JVM runs out of memory. On the other hand, exceptions are mainly caused by the application. For instance, Null Pointer Exception happens when an app tries to get through a null object.

Q3: Why do we need exception handling in Java?

A: If there is no try and catch block while an exception occurs, the program will terminate. Exception handling ensures the smooth running of a program without program termination.

Q4: What are checked and unchecked exceptions in Java?

A: Checked: Occur during the compilation. Here, the compiler checks whether the exception is handled and throws an error accordingly.

Unchecked: Occur during program execution. These are not detectable during the compilation process.

Q5: What is the difference between the throw and throws keywords in Java?

A: We use the “throws” keyword to declare the exception with the method declaration. And “throw” is used to forcibly throw the exception, while the try-catch blocks is used to handle the exceptions thrown by the code.

Q6: How do you handle checked exceptions?

A: Checked Exceptions can be handled by either using a try-catch block or by using the throws clause in the method declaration. If these exceptions are not handled properly, then the program would fail to compile.

Q7: What is String in Java?

A: Strings, one of the most common objects used in Java programming, are essentially sequences of characters.

Q8: How to declare a string in Java?

A: By string literal: Double quotes are used to create Java String literals. Eg: String name="Java";

By new keyword: Keyword "new" is used to create a Java string. Eg: String str=new String ("Java");

Q9: Why is String immutable in Java?

A: Immutable objects mean they can't be changed or altered once they've been created. The String is immutable in Java because of many reasons like security, caching, synchronization and concurrency, and class loading.

Q10: How to check if a String is empty or not?

A: Use the isEmpty() method

Q11: What is StringBuffer is Java?

A:

A StringBuffer is a mutable object, meaning it can be changed, but the string is an immutable object, so it cannot be changed once it has been created.

Q12: Difference between == and equals()

A:

The == operator can be used for comparing references (addresses) and the .equals() method can be used to compare content. To put it simply, == checks if the objects point to the same memory location, whereas .equals() compares the values of the objects.

Q13: Name a few methods in String and explain what they do?

A:

split(): Split/divide the string at the specified regex.

compareTo(): Compares two strings on the basis of the Unicode value of each string character.

compareToIgnoreCase(): Similar to compareTo, but it also ignores case differences.

length(): Returns the length of the specified string.

substring(): Returns the substring from the specified string.

equalsIgnoreCase(): Compares two strings ignoring case differences.

contains(): Checks if a string contains a substring.

trim(): Returns the substring after removing any leading and trailing whitespace from the specified string.

charAt(): Returns the character at specified index.

toLowerCase(): Converts string characters to lower case.

toUpperCase(): Converts string characters to upper case.

concat(): Concatenates two strings.

**Java – Inheritance,Abstract Classes & Interfaces – Programs**

Q1: O/p of following program,?

class Base {

public void show() {

System.out.println("Base::show() called");

}

}

class Derived extends Base {

public void show() {

System.out.println("Derived::show() called");

}

}

public class Main {

public static void main(String[] args) {

Base b = new Derived();

b.show();

}

}

A. Base::show() called

B. Derived::show() called

Q2: O/p of following program?

class Base {

final public void show() {

System.out.println("Base::show() called");

}

}

class Derived extends Base {

public void show() {

System.out.println("Derived::show() called");

}

}

class Main {

public static void main(String[] args) {

Base b = new Derived();

b.show();

}

}

A. Base::show() called

B. Derived::show() called

C. Compile Error

D. Runtime Error

Q3: O/p of following?

class Base {

public static void show() {

System.out.println("Base::show() called");

}

}

class Derived extends Base {

public static void show() {

System.out.println("Derived::show() called");

}

}

class Main {

public static void main(String[] args) {

Base b = new Derived();

b.show();

}

}

A. Base::show() called

B. Derived::show() called

C. Compile Error

D. Runtime Error

Q4: What is the problem with following code?

class X

{

//Class X Members

}

class Y

{

//Class Y Members

}

class Z extends X, Y

{

//Class Z Members

}

Q5: O/p of following program?

class X

{

{

System.out.println(1);

}

}

class Y extends X

{

{

System.out.println(2);

}

}

class Z extends Y

{

{

System.out.println(3);

}

}

public class Main

{

public static void main(String[] args)

{

Z c = new Z();

}

}

Q6: O/p of following:

class X

{

static

{

System.out.println("THIRD");

}

}

class Y extends X

{

static

{

System.out.println("SECOND");

}

}

class Z extends Y

{

static

{

System.out.println("FIRST");

}

}

public class Main

{

public static void main(String[] args)

{

Z c = new Z();

}}

Q7: O/P of following program?

class X

{

public X()

{

System.out.println("Class X Zonstructor");

}

}

class Y extends X

{

public Y()

{

System.out.println("Class Y Zonstructor");

}

}

class Z extends Y

{

public Z()

{

System.out.println("Class Z Zonstructor");

}

}

public class Main

{

public static void main(String[] args)

{

Z z = new Z();

}

}

Q8: What is the issue in this program?

abstract class AbstractClass

{

abstract void abstractMethod()

{

System.out.println("First Method");

}

}

Q9: What is the issue here?

abstract class A

{

abstract int add(int a, int b);

}

class B extends A

{

}

Q10: O/p of following program?

abstract class X

{

public X()

{

System.out.println("ONE");

}

abstract void abstractMethod();

}

class Y extends X

{

public Y()

{

System.out.println("TWO");

}

@Override

void abstractMethod()

{

System.out.println("THREE");

}

}

public class Main

{

public static void main(String[] args)

{

X x = new Y();

x.abstractMethod();

}

}

Q11:

**Java – Inheritance,Abstract Classes & Interfaces - Theory**

Q1: What is inheritance in Java?

Answer:

Inheritance, is one of three object oriented concepts, which helps to separate out common data and behavior (or member variables and methods) from multiple related classes. Inheritance greatly helps in simplifying the code, enhancing its the re-usability and maintainability.

Q2: What are static variables?

Answer:

The static keyword is used to create variables that will exist independently of any instances created for the class. Only one copy of the static variable exists regardless of the number of instances of the class.

Static variables are also known as class variables. Local variables cannot be declared static.

Q3: What are static methods?

Answer:

The static keyword is used to create methods that will exist independently of any instances created for the class.

Static methods do not use any instance variables of any object of the class they are defined in. Static methods take all the data from parameters and compute something from those parameters, with no reference to variables.

Q4: What are final variables?

Answer: A final variable can be explicitly initialized only once. A reference variable declared final can never be reassigned to refer to an different object.

However, the data within the object can be changed. So, the state of the object can be changed but not the reference.

With variables, the final modifier often is used with static to make the constant a class variable.

Q5: What are final methods?

Answer:

A final method cannot be overridden by any subclasses. As mentioned previously, the final modifier prevents a method from being modified in a subclass.

The main intention of making a method final would be that the content of the method should not be changed by any outsider.

Q6: What are final classes?

Answer:

The main purpose of using a class being declared as final is to prevent the class from being subclassed. If a class is marked as final then no class can inherit any feature from the final class.

Q7: What are Abstract Classes in Java? Can you give an example?

Answer:

An abstract class can never be instantiated. If a class is declared as abstract then the sole purpose is for the class to be extended.

Eg:

abstract class Caravan {

private double price;

private String model;

private String year;

public abstract void goFast(); // an abstract method

public abstract void changeColor();

}

Q8: What are Abstract methods?

Answer:

An abstract method is a method declared without any implementation. The methods body (implementation) is provided by the subclass. Abstract methods can never be final or strict.

Any class that extends an abstract class must implement all the abstract methods of the super class, unless the subclass is also an abstract class.

The abstract method ends with a semicolon. Example: public abstract sample();

Q9: A class contains one abstract method, should the class be marked as abstract?

Answer:

Yes. If a class contains one or more abstract methods, then the class must be declared abstract whereas an abstract class does not need to contain abstract methods.

Q10: What is method overriding?

Answer:

Method overriding is a powerful concept supported by Java, which helps in changing the behavior of the method implemented in the super-class.

Q11: What are Interfaces in Java and how to declare them?

Answer:

Interfaces is a powerful concept of Java which helps in achieving multiple inheritance to certain extent.

Interface is similar to a abstract class except that all the methods in it are abstract. We can not include the method body for any of the methods in interface i.e. we can not define concrete methods in interface.

Eg:

interface interface-name

{

return-type method-name-1(parameters);

return-type method-name-2(parameters);

}

Q12: How do we use interface in an concrete class ? Give an example.

Answer:

We need to use the keyword implements.

Eg: If IBowler is an interface in this way:  
interface IBowler{

void bowl();

}

Then a class need to implement the interface in this manner:

class Bowler implements IBowler{

void bowl(){

System.out.println(“In bowl method”);

}

}

**Java – Classes and Objects**

Q1: What is a Class and Object in Java?

Answer:

A class is a user-defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. Using classes, you can create multiple objects with the same behavior instead of writing their code multiple times.

An object is a basic unit of Object-Oriented Programming that represents real-life entities. A typical Java program creates many objects for the same class.

Q2: What are the pillars of OOPs?

Answer: Abstraction,Encapsulation,Inheritance,Polymorphism

Q3: What is Abstraction in Java? Can you give an example?

Answer:

Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or non-essential units are not displayed to the user.

Eg: A person driving a car knows that pressing the accelerators will increase the car speed or applying brakes will stop the car, but he/she does not know how on pressing the accelerator, the speed is actually increasing. He/She does not know about the inner mechanism of the car or the implementation of the accelerators, brakes etc. in the car. This is what abstraction is.

Q4: What is Encapsulation in Java? Can you give an example?

Answer: It is defined as the wrapping up of data under a single unit. It is the mechanism that binds together the code and the data it manipulates.

Eg: If combinations of medicine are variables and methods then the capsule will act as a class and the whole process is called Encapsulation

Q5: What is Inheritance in Java? Can you give an example?

Answer:

Inheritance is an important pillar of OOP (Object Oriented Programming). It is the mechanism in Java by which one class is allowed to inherit the features (fields and methods) of another class.

Real-life example – Parent-child. A child inherits certain features from parents.

Q6: How to create an object in Java?

Answer: By using the new keyword. Eg: Box mybox = new Box();

Q7: What is a Constructor in Java?

Answer: A constructor initializes an object immediately upon creation. It has the same name as the class in which it resides and is syntactically similar to a method.

Q8: What is a parameterized constructor in Java?

Answer: Constructor that takes certain parameters to assign the attributes of a class is called a parameterized constructor

Q9: What is “this” keyword in Java?

Answer: Sometimes a method will need to refer to the object that invoked it. b. To allow this, Java defines the this keyword. this can be used inside any method to refer to the current object.

Q10: How do we create a method in Java?

Answer: This is the general form of a method:

type name(parameter-list) { // body of method

}

Q11: What is overloading in Java? Can you give an example?

Answer:

When we have more than one method with same name, but different parameters, then Java decides which one to call at run time.

Eg:

int add(int num1,int num2);

double add(double num1, double num2);

Here both methods have same name add, but have different parameters

Q12: What is call by value and call by reference in Java?

Answer:

Call By Value: In Java, when you pass a primitive type to a method, it is passed by value. Thus, what occurs to the parameter that receives the argument has no effect outside the method.

Call By Reference: When you create a variable of a class type, you are only creating a reference to an object. Thus, when you pass this reference to a method, the parameter that receives it will refer to the same object as that referred to by the argument. If any attribute of the reference is changed,we see the change in the object attributes as well.

Q13: How many constructors are there in below class?

class Test{

int i;

int j;

Test(){

}

Test(int i, int j){

}

void m1(){

}

Test(int i){

}

}

Answer : 3

Q14: O/p of following program?

class T {

int t = 20;

}

class A {

public static void main(String args[]) {

T t1 = new T();

System.out.println(t1.t);

}

}

Answer: 20

Q15: O/p of following program

class T {

int t = 20;

T() {

t = 40;

}

}

class A {

public static void main(String args[]) {

T t1 = new T();

System.out.println(t1.t);

}

}

Answer: 40

Q16:

**Java – Arrays**

Q1: What are Arrays in Java?

Answer: An array is, essentially, a list of like-typed variables. To create an array, you first must create an array variable of the desired type. The general form of a one-dimensional array declaration is

type var-name[ ];

Q2: What is ArrayOutOfBoundsException wrt to Arrays?

Answer: It occurs when a program attempts to access an invalid index in an array i.e. an index that is less than 0, or equal to or greater than the length of the array

Q3: Which index do array elements start from?

Answer: 0

Q4: Output of following

class A{

public static void main(String[] args){

int[] rollNumber = { 23, 17, 20, 29, 30 };

for (int temp : rollNumber)

System.out.print(temp+" ");

}

} }

Answer: Error – We are missing a bracket { at the for loop.

Q5: What is the length of the following array: byte[] data = { 12, 34, 9, 0, -62, 88 };

Answer: 6

Q6: O/p of following program:  
class A{

public static void main(String[] args){

int i[] = new int[10];

System.out.println(i[10]);

} }

Answer: Exception. We are trying to access index 10, the array will have a max index of 9 since its length is 10.

Q7: Given, int [] nums = {2, 3, 5, 8, 9, 11};

How would you access the fourth element in nums?

Answer: a[3] – This will give the fourth element.

Q8: O/p of following program:

class A{

public static void main(String[] args){

int [] arr1= {1, 2, 3};

int [] arr2 = {1, 2, 3};

if(arr1[0] == arr2[0])

System.out.println("Same");

else

System.out.println("Not same");

} }

Answer: Same

Q9: Which is correct declaration of a 2d array?

int intArray1[][]; // A

int[][] intArray2; // B

Answer: Both are correct.

Q10: O/p of following program?

class A{

public static void main(String[] args){

int i[] = new int[10];

i[0]= 8;

System.out.println(i[0]+i.length);

} }

Answer: 18

**Java – Operators**

Q1: Give examples of Arithemetic operators

Answer: Some arithmetic operators are +, -, ++, -- etc

Q2:What is output of following program  
class A{

public static void main(String[] args){

int a = 10;

int b=20;

System.out.println(a++);

System.out.println(b++);

System.out.println(a);

System.out.println(b);

} }

Answer:

10

20

11

21

Q3: What is the output of following program?

class A{

public static void main(String[] args){

int a = 10;

int b=20;

a += 7;

b -= 3;

System.out.println(a);

System.out.println(b);

} }

Answer:

17

17

Q4: What is the output of following program?

class A{

public static void main(String[] args){

int a = 10;

int b=20;

System.out.println(a&b);

System.out.println(a|b);

} }

Answer:

0

30

Q5: What is the output of following program?

class A{

public static void main(String[] args){

int a = 10;

int b=20;

System.out.println(a>>1);

System.out.println(b<<2);

}}

Answer:

0

30

Q6: O/p of below program?

class A{

public static void main(String[] args){

int a = 10;

int b=20;

int c = a < b ? a > 8 ? 9 : 4 : 5;

System.out.println(c);

}}

Answer:

9

Q7: O/p of this- System.out.println(6 + 4 \* 5 + 2);

Answer: 28

Q8: O/p of this - System.out.println( (8 + 1)\* 4 + 5 \* 3);

Answer: 51

Q9: Write the if statement to print if a number is even or odd. (Use if else)

Answer:

If(number%2==0){

System.out.println(“Even”);

}

else{

System.out.println(“Odd”);

}

Q10: Program to find factorial of a number

Answer: Discussed in class in details

**Java – Basic Questions**

Q1: What is Java?

Answer: Java is a high-level, class-based, object-oriented ,platform independent programming language

Q2: Can you give a few buzz words of Java?

Answer : Fe buzz words of Java are – Simple, Secure, Portable, Object-Oriented, Robust etc.

Q3: From which method does Java program processing start?

Answer: It starts from the main method.

Q4: How to compile a program in Java?

Answer: Use the command javac <file\_name>.java

Q5: When we compile a java program which file is generated?

Answer: When we compile a program, a .class file is generated.

Q6: How to run a Java Program?

Answer: Use command java <name\_of\_the\_class>

Q7: Can you write a simple program to print Hello World using Java?

Answer: public class MyFirstJavaProgram {

public static void main(String []args) {

System.out.println("Hello World"); // prints Hello World

}

}

Q8: What happens at compile time?

Answer: At compile time, the Java file is compiled by Java Compiler and converts the Java code into bytecode.

Q9: What is JVM?

Answer: JVM (Java Virtual Machine) is an abstract machine that enables your computer to run a Java program.

When you run the Java program, Java compiler first compiles your Java code to bytecode. Then, the JVM translates bytecode into native machine code

Q10: Why is Java platform independent?

Answer: It's because when you write Java code, it's ultimately written for JVM but not your physical machine (computer). Since JVM executes the Java bytecode which is platform-independent, Java is platform-independent.

Q11: What is JRE?

Answer: JRE (Java Runtime Environment) is a software package that provides Java class libraries, Java Virtual Machine (JVM), and other components that are required to run Java applications.

Q12: What is JDK?

Answer: JDK (Java Development Kit) is a software development kit required to develop applications in Java. When you download JDK, JRE is also downloaded with it.

In addition to JRE, JDK also contains a number of development tools (compilers, JavaDoc, Java Debugger, etc).

Q13: What are the primitive types in Java?

Answer: We have 8 primitive types in Java, they are:   
byte, short, int, long, char, float, double, and boolean.

Q14: List a few Java keywords

Answer: There are many Java keywords, some of them are for,if,else,continue,break etc.

**Commands in Git**

Q1: What are branches in Git?

Answer : A branch is a version of the repository that diverges from the main working project. A git project can have more than one branch.

Q2 : What is the master branch in Git?

Answer : The master branch is the default branch in Git. It is instantiated when first commit made on the project. A repository can have only one master branch. Some companies may also use the name main for this starting branch.

Q3 : How to create a branch?

Answer : Use command git branch <branch\_name>

Eg: To create a dev branch we can create using, git branch dev

Q4 : How to see all branches of a repository ?

Answer: Use command: git branch

Q5 : How to move to a particular branch?

Answer : Use command: git checkout <branch\_name>

Eg: To move to dev branch, the command is git checkout dev

Q6: How to create and checkout a branch at the same time?

Answer: Use command: git checkout -b <branch\_name>

Q7: How to merge a branch to the other branch?

Answer : Use command: git merge dev

If you are on master branch, this will merge the dev branch changes to the master branch

Q8: What is the use of the .gitignore file?

Answer: If there are any files in the project that we do not want to track and do not want them to show up in the git status command, we can add such files to the .gitignore file.

Q9: How to see what changes we have made to a file with what is currently present in the file?

Answer: Use command git diff <file\_name>

Q10: What is GitHub?

Answer: GitHub, Inc. is an Internet hosting service for software development and version control using Git.

Q11 : What does git remote command do?

Answer: It helps to create,view and delete connections to other repositories

Q12 : What does git push command do?

Answer : It updates remote refs along with associated objects.

Eg: Once you make changes to your branch on local and want to push it to git hub, you could use this command: git push -u origin master

Q13: What does git pull command do?

Answer : The git pull command is used to fetch and download content from a remote repository and immediately update the local repository to match that content.

Q14 : How to clone a Git Repository?

Answer : Use the command git clone <URL>

Q15 : How to check to which repository you are connected to from your local machine?

Answer: Use command git remote -v

**Git – Questions and Answers**

Q1 : What is GIT ?

Answer: Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers collaboratively developing source code during software development.

Q2 : Why is GIT used?

Answer: GIT is used for Synchronous development, to increase team speed and productivity. It is also used across many industries and has become an industry standard.

Q3 : What is the command used to set Git config?

Answer : To set username use : **git config –global user.name <name>**

To set email use: **git config –global user.email <email>**

Q4 : What does git init command do?

Answer: It is used to initialize a new or an empty repository to a Git repository.

Q5 : How do add a file to the staging area?

Answer: Use git add <file\_name> to add a file to a staging area.

Q6 : What does git commit command do?

Answer: It records the file(s) permanently in the version history.

Q7 : Which command is used to see the commit history?

Answer : The git log command is used to see the commit history

Q8 : What does the git status command do?

Answer: It displays the state of the working directory and the staging area.

Q9: How to add all files to the staging area?

Answer : Use the command **git add .**

Q10: How to go back to a previous commit?

Answer: Use the command git checkout <commit\_id>