SAQ:

1. Define Encapsulation?

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as **data hiding**.

2. Define Inheritance?

Inheritance can be defined as the process where one class acquires the properties (methods and fields) of another.

The class which inherits the properties of other is known as subclass (derived class, child class) and the class whose properties are inherited is known as superclass (base class, parent class).

3. Define Polymorphism?

Polymorphism is the ability of an object to take on many forms.

There are two types of polymorphism in java: compile time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

4. Briefly explain history of Java?

**James Gosling** initiated Java language project in June 1991 for use in one of his many set-top box projects. The language, initially called ‘Oak’ after an oak tree that stood outside Gosling's office, also went by the name ‘Green’ and ended up later being renamed as Java, from a list of random words.

5. Describe a constant?

Constants are variables that are declared as public/private, final, and static. Constant variables never change from their initial value.

For example, the following variable declaration defines a constant named PI, whose value is an approximation of pi (the ratio of the circumference of a circle to its diameter):

static final double PI = 3.141592653589793;

6. Define Scope and life time of variables?

Scope refers to the lifetime and accessibility of a variable. How large the scope is depends on where a variable is declared.

For example, if a variable is declared at the top of a class then it will accessible to all of the [class methods](http://java.about.com/od/workingwithobjects/a/constructor.htm). If it’s declared in a method then it can only be used in that method.

7. List types of operators?

Java provides a rich set of operators to manipulate variables. All Java operators are classified into the following groups:

* Arithmetic Operators
* Relational Operators
* Bitwise Operators
* Logical Operators
* Assignment Operators

? Operator

8. Define implicit type casting?

A data type of lower size (occupying less memory) is assigned to a data type of higher size. This is done implicitly by the JVM. The lower size is widened to higher size. This is also named as automatic type conversion.

9. Define explicit type casting?

A data type of higher size (occupying more memory) cannot be assigned to a data type of lower size. This is not done implicitly by the JVM and requires **explicit casting**; a casting operation to be performed by the programmer. The higher size is narrowed to lower size.

10. Define enumerated data type?

An enum type is a special data type that enables for a variable to be a set of predefined constants. The java enum constants are static and final implicitly.

11. Define an array?

An array is a group of like-typed variables that are referred to by a common name. Arrays of any type can be created and may have one or more dimensions. A specific element in an array is accessed by its index. Arrays offer a convenient means of grouping related information.

12. Define this reference?

In java, “this” is a **reference variable** that refers to the current object.

“this” keyword can be used to refer current class instance variable.

13. Define Constructor?

**Constructor in java** is a special type of method that is used to initialize the object. **Java constructor** is invoked at the time of object creation.

14. Define Recursion?

Recursion in java is a process in which a method calls itself continuously. A method in java that calls itself is called recursive method.

Ex: Factorial of a number, Fibonacci series.

15. Define Garbage collection?

In java, garbage means unreferenced objects.

Garbage Collection is process of reclaiming the runtime unused memory automatically. In other words, it is a way to destroy the unused objects.

LAQ:

1. Distinguish between Procedure Oriented Programming and Object Oriented Programming?



1. Describe features (buzzwords) of Java Programming Language?

**Simple**:

Java is easy to learn and its syntax is quite simple, clean and easy to understand. The confusing and ambiguous concepts of C++ are either left out in Java or they have been re-implemented in a cleaner way.

Eg : Pointers and Operator Overloading are not there in java but were an important part of C++.

**Object-Oriented**: Java is purely object oriented.

Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior. This feature allows for re-usability of code and maintainability.

**Platform independent**: A platform is the hardware or software environment in which a program runs.

Java provides software-based platform. Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, Mac/OS etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere(WORA).



Fig: Java Compilation Process

**Secure**:

Java program always runs in Java runtime environment with almost null interaction with system OS, hence it is virus free and tamper free with more security.

**Robust**: Robust means strong.

Java improved were Memory Management and mishandled Exceptions by introducing automatic **Garbage Collector** and **Exception Handling**.

**Architecture neutral**:

Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.

**Portable**:

Being architecture-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary, which is a POSIX subset.

**Dynamic**:

Java programs carry with them substantial amounts of run-time type information that is used to verify and resolve accesses to objects at run time.

**Interpreted** :

Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.

**High Performance**:

With the use of Just-In-Time compilers, Java enables high performance.

**Multithreaded**:

With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.

**Distributed**:

Java is designed for the distributed environment of the internet.

1. What is this reference in java?Explain its usage.

In java, “this” is a **reference variable** that refers to the current object.

## Usage of java “this” keyword:

1. this keyword can be used to refer current class instance variable.
2. this() can be used to invoke current class constructor.
3. this keyword can be used to invoke current class method (implicitly)
4. this can be passed as an argument in the method call.
5. this can be passed as argument in the constructor call.
6. this keyword can also be used to return the current class instance.

4. Write about the method overloading with an example?

When a class has two or more methods by the same name but different parameters, it is known as method overloading. It is different from overriding. In overriding, a method has the same method name, type, number of parameters, etc.

Different ways to overload the method

|  |
| --- |
| There are two ways to overload the method in java |

1. By changing number of arguments
2. By changing the data type

## METHOD OVERLOADING BY CHANGING THE NO. OF ARGUMENTS

## class Calculation{

##   void sum(int a,int b){System.out.println(a+b);}

##   void sum(int a,int b,int c){System.out.println(a+b+c);}

##

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

##   Calculation obj=new Calculation();

##   obj.sum(10,10,10);

##   obj.sum(20,20);

##

##   }

## }

Output:30

 40

5. Write about the constructor overloading with an example?

## Constructor Overloading in Java

|  |
| --- |
| Constructor overloading is a technique in Java in which a class can have any number of constructors that differ in parameter lists. The compiler differentiates these constructors by taking into account the number of parameters in the list and their type. |

## CONSTRUCTOR OVERLOADING

## class Student5{

##     int id;

##     String name;

##     int age;

##     Student5(int i,String n){

##     id = i;

##     name = n;

##     }

##     Student5(int i,String n,int a){

##     id = i;

##     name = n;

##     age=a;

##     }

##     void display(){System.out.println(id+" "+name+" "+age);}

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

##     Student5 s1 = new Student5(111,"Karan");

##     Student5 s2 = new Student5(222,"Aryan",25);

##     s1.display();

##     s2.display();  }  }

Output:

111 Karan 0

222 Aryan 25

6. What is an array? Explain different types of array with example?

An array is a group of like-typed variables that are referred to by a common name. Arrays of any type can be created and may have one or more dimensions. A specific element in an array is accessed by its index. Arrays offer a convenient means of grouping related information.

**One-Dimensional Arrays**

A one-dimensional array is, essentially, a list of like-typed variables.

The general form of a one-dimensional array declaration is:

type var-name[ ];

Here, type declares the base type of the array.

#### Example:

#### class AutoArray {

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

#### int month\_days[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31,

#### 30, 31 };

#### System.out.println("April has " + month\_days[3] + " days.");

#### }

#### }

**Multidimensional Arrays**

In Java, multidimensional arrays are actually arrays of arrays.

For example, the following declares a two dimensional array variable called **twoD**.

int twoD[][] = new int[4][5];

This allocates a 4 by 5 array and assigns it to **twoD**.

#### CODE FOR ARRAY (TWO-DIMENSIONAL)

#### class TwoDArray {

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

## int twoD[][]= new int[4][5]; // Demonstrate a two-dimensional array.

#### int i, j, k = 0;

#### for(i=0; i<4; i++)

#### for(j=0; j<5; j++) {

#### twoD[i][j] = k;

#### k++;

#### }

#### for(i=0; i<4; i++) {

#### for(j=0; j<5; j++)

#### System.out.print(twoD[i][j] + " ");

#### System.out.println();

#### }

#### }

#### }

This program generates the following output:

0 1 2 3 4

5 6 7 8 9

10 11 12 13 14

15 16 17 18 19

7. Define static keyword with its application to variable, method and block?

# Java Static Methods and Variables

Types of Variable

1. static variables
2. static methods
3. static blocks of code

### Java static variable

* It is a variable which **belongs to the class** and **not**to **object**(instance)
* Syntax : <**class-name>.<variable-name>**

### Java Static Method

* It is a method which **belongs to the class**and **not**to the **object**(instance)
* Syntax : <**class-name>.<method-name>**

## static block

The static block, is a block of statement inside a Java class that will be executed when a class is first loaded in to the JVM

8. Explain switch and break statements with an example program?

**Switch Statement in Java**

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

**Syntax**

The syntax of enhanced for loop is:

switch(expression){

case value :

//Statements

break; //optional

case value :

//Statements

break; //optional

//You can have any number of case statements.

default : //Optional

//Statements

}

## SWITCH

#### public class Test {

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

#### char grade = 'C'; //char grade = args[0].charAt(0);

#### switch(grade) {

#### case 'A' :

#### System.out.println("Excellent!");

#### break;

#### case 'B' :

#### System.out.println("Very Good");

## break;

#### case 'C' :

#### System.out.println("Well done");

#### break;

#### case 'D' :

#### System.out.println("You passed");

#### case 'F' :

#### System.out.println("Better try again");

#### break;

#### default :

#### System.out.println("Invalid grade"); }

#### System.out.println("Your grade is " + grade); } }

This will produce the following result:

$ java Test

Well done

Your grade is a C

9. Explain for loop with an example program?

**for Loop in Java**

A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to be executed a specific number of times.

**Syntax**

The syntax of a for loop is:

for(initialization; Boolean\_expression; update)

{

//Statements

}

**CODE OF THE FOR LOOP IN JAVA.**

#### public class Test {

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

#### for(int x = 10; x < 20; x = x+1) {

#### System.out.print("value of x : " + x );

#### System.out.print("\n");

#### } } }

This will produce the following result:

value of x : 10

value of x : 11

value of x : 12

value of x : 13

value of x : 14

value of x : 15

value of x : 16

value of x : 17

value of x : 18

value of x : 19

10. Explain a selection statement with an example program?

**If Statement in Java**

An **if** statement consists of a Boolean expression followed by one or more statements.

**Syntax**

Following is the syntax of an if statement:

if(Boolean\_expression)

{

//Statements will execute if the Boolean expression is true

}

**C 1.15 CODE FOR “IF” STATEMENT**

#### public class Test {

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

#### int x = 10;

#### if( x < 20 ){

#### System.out.print("Value of x is less than 20");

#### } } }

This will produce the following result:

Value of x is less than 20