## CH 1: INTRODUCTION TO OBJECT ORIENTED PROGRAMMING CONCEPTS

* 1. What are the two types of programming languages?

Ans. Low level languages and high level languages.

* 1. What are the programming language generations?

Ans. First generation language (machine language), second generation language( assembly language), third generation language(BASIC, COBOL, C),fourth generation language (SQL, WAVE) and the fifth generation language (Prolog, Mercury).

* 1. What do you mean by programming paradigm?

Ans. It is an approach to solve a given problem by writing a set a instructions in a programming language. It refers to the organizing principles of a program.

* 1. What are the different programming paradigms?

Ans: Procedural or modular programming and Object oriented Programming.

* 1. What is Object Oriented Programming?

Ans: The object oriented programming approach focuses a problem in terms of classes and objects. The OOP determines the classes and objects needed for the solution of a problem and provides a full set of operations for each class.

* 1. List the concepts of OOPs.

Ans: The concepts of OOPs are:

* Data Abstraction
* Encapsulation
* Modularity
* Inheritance
* Polymorphism
  1. Write the differences between Procedural Programming and Object Oriented Programming.

Ans: Procedural programming aims more at procedures. The emphasis is a doing things rather than the data being used. In procedural Programming paradigm data are shared among all the functions participating thereby risking data safety and security. Object Oriented Programming is based on principles of data hiding, abstraction, inheritance and polymorphism. It implements programs using classes and objects, In OOP‟s data and procedure both given equal importance. Data and functions are encapsulated to ensure data safety and security.

* 1. What is an abstraction?

Ans: The act of representing the essential features without including the background details or explanation. E.g. in a switchboard we press the switches according to our requirements without knowing the circuitry.

* 1. What is encapsulation?

Ans:The wrapping up of data and functions into a single unit called a class in known as encapsulation. E.g In a medicine capsule all the constituents are encapsulated inside the capsule. Doctor administers the intake of the entire capsule.

* 1. What is inheritance and how it is useful in Java. Give examples. [2008]

Ans: It is process by which objects of one class acquire the properties of objects of another class. Inheritance supports the concepts of hierarchical representation. In OOP the concepts of inheritance provides the idea of reusability.

e.g. A class CAR inherits some property from the class Automobiles, which in turn inherits properties from Vehicle class. The capability to pass down properties and so allows us to describe things efficiently.

* 1. What role does polymorphism play as java feature?

Ans: It means the ability for a message or data to take more than one form. For example, an operation, many types of data used in the operation.

E.g. a child behaves like a student in school, customer in a shopping mall, passenger in a bus and like a son/daughter at home. Same person behave differently in various circumstances

**Ch 2: INTRODUCTION TO JAVA**

1. What was Java originally known as? Who developed Java?

Ans. Java was originally known as OAK. It was developed by James Gosling .

1. What are the characteristics of Java?

Ans: The characteristics of Java as follows:

1. It is an object oriented language, simple and robust.
2. Java program is both compiled and interpreted.
3. Write once run anywhere (WORA). The java programs are written once, but can be run on different platform without making changes.
4. Has enhanced security features. v)Platform independent.
5. What is the difference between source code and object code?

Ans. The program code written by the programmer is called the source code.

The source code in translated into the machine understandable code called as object code or the native executable code.

1. How does ordinary compilation takes place?

Ans. The process of converting the source code into machine code is called compilation. The compilation produces the machine code. For different platforms different machine codes are produce.

e.g Windows compiler produces an object code that run in Windows computer only.

1. Briefly explain the compilation process of Java source code.

Ans. Compilation in java:

1. Java programs are written in “.java” files and are compiled by the compiler to produce the bytecode (.class file)
2. The JVM is a virtual machine which reads the byte code and interprets into machine code(object code)
3. Explain the two types of Java programs.

Ans: Two types of Java programs are Applets and Stand alone applications.

* 1. Applets – small programs that are embedded in web pages and are run on the viewer‟s machine by Java capable browsers. Used for games and simulation programs.
  2. Standalone application- includes the software applications that do not require low level operating system or hardware access. This includes most desktop applications like word processors and spreadsheets

1. What is a bytecode?

Ans: Bytecode is an intermediate code produce after compilation of Java source code. Bytecode is a set of pseudo machine language instructions that are understood by the JVM (Java Virtual Machine) and are independent of the underlying hardware (platform independent).

1. What do you understand by JVM?

Ans: JVM or Java Virtual Machine is an abstract machine designed to be implemented on top of existing processors. It hides the underlying OS from Java application. Programs written in Java are compiled into Java byte-code, which is then interpreted by a special java Interpreter for a specific platform. This Java interpreter is known as Java Virtual Machine (JVM).

1. What is JDK (Java Development Kit)?

Ans: The Java development kit comes with a collection of tools that are used for developing and running java programs.

1. What are Java APIs?

Ans: The Java APIs (Application Program Interface) consist of libraries of pre-compiled code that programmers can use in their application.

1. What do you know about BlueJ?

Ans: BlueJ is a Java development environment. It is an IDE (Integrated Development Environment), which includes an editor a debugger and a viewer.

1. How you create, compile and execute a program in Java or BlueJ? Explain your answer?

Ans: Create: Click on New Class button from BlueJ editor, then type the class name a program icon will be created. Double click on it, a program editor will be open, erase the code and type your program coding.

Compile: click the Compile button on the left of the window or right click on the class icon and select compile from the menu options.

Execute: Right click on the class icon and select new class name option. A dialogue box appears type the name of the object. An object icon will be created at the bottom. Right click on the object icon and select the method we want to execute.

1. Why do we use main( )?

Ans. In a Java standalone program the execution begins from the main( ). The main( ) is the entry point in Java and all other functions are called from this method.

## CH 3: ELEMENTARY CONCEPT OF OBJECTS AND CLASSES

1. Define the terms:
2. Object- an object is an identifiable entity with some characteristics/properties/attribute, state and behavior.
3. Attributes- the properties defining an object are termed as attributes or characteristics of an object.
4. State – the state of an object includes all the static properties plus the current values of these properties.
5. Class- a class is a blueprint defining the structure( characteristics and behavior) of objects on which actual objects are created.
6. How do objects communicate with each other?

Ans. Objects communicate with each other through message passing.

1. How can you implement abstraction in programming terms?

Ans. To implement abstraction in software terms we define a class. There can be many classes for an entity.

1. Classes are user defined type. Explain.

Ans. To represent an abstraction in programming terms, a user defined type is created based on which multiple objects of same type can be created. Thus in software terms, a class represents a type with certain specifications. Once a class in created, multiple objects of this class type can be created.

1. How are classes and objects implemented in software terms?

Ans. In software terms, classes and objects are implemented by representing the characteristics/properties/attributes as data members and behavior through methods/member-functions.

1. “Object is an instance of a class”, Explain

Ans: Object of a class contains data and functions provided in a class. It possesses all the features of a class. Hence object is termed as instance of a class.

1. State two differences between fundamental and user-defined data type. Ans.

|  |  |
| --- | --- |
| Comparison of Fundamental & User defined data type. | |
| Fundamental data type | User defined data type |
| These are inbuilt data type provided by the Java Language. | These are data types created by the user  using fundamental or user defined data type or both. |
| The size of it is fixed. | The size of different user-defined data  type depends upon the size of the individual components of it. |
| These data types are available in all parts of a program within a class. | These data types are available only as specified by the access specifiers. |

1. State two differences between a class and an object. Ans.

|  |  |
| --- | --- |
| A class | An object |
| Class is a blueprint or template from which objects are created. | Object is an instance of a class. |
| Class is a group of similar objects. | Object is a real world entity such as pen,  laptop, mobile, bed, keyboard, mouse, chair etc. |
| Class is a logical entity. | Object is a physical entity. |

1. Why is a class called a user-defined data type?

Ans. Primitive data types are the general and fundamental data types that we have in Java and those are byte, short, int, long, float, double, char, boolean, etc., User defined data types are those that user / programmer himself defines.

1. What is an access specifier? Which two access specifier is used in a class declaration?

Ans. Access modifiers (or access specifiers) are keywords in object-oriented languages that set the accessibility of classes, methods, and other members. Access modifiers are a specific part of programming language syntax used to facilitate the encapsulation of components.

The public and default access specifier is used in a class declaration.

1. Why is a class called an object factory?

Ans. A class is called an object factory because objects are created from a class. An object is an instance of a class.

The following statements create two objects s1 and s2 of the class Student. Student s1 = new Student();

Student s2 = new Student();

So, we have a single class Student but we can create as many objects as we want (like s1, s2, etc.) from that single class.

This is similar to what happens in a factory. Consider a factory which produces car. They have only a single design of a car but produce multiple cars from that single design.

Things are similar in the world of classes and objects. There is a single definition of a particular class (like Student) but we can produce many Student objects (like s1, s2) from that single class.

1. State two rules you should follow for naming a class.

Ans. While using names for a class the following set of rules are to be kept in mind.

* 1. It can have any alphabet (capital or small), digits, underscore and dollar sign characters. For example, *a, b, cat, mat123, cost\_price, Topaz$* are all example of valid identifier.
  2. It should not begin with digits or should not contain any special character. For example *2ab, ab#c, top@, etc.*, are invalid identifiers as it either begins with a digit or contain special characters (like #, @).
  3. It cannot have a space between it. For example, *Simple Interest* or *Selling Price* are invalid class-names as it contains a space.
  4. It must not be a *keyword*. For example, *for, while, do* are invalid class-names as they are keywords and are assigned some special function for the language.
  5. It can be of any length. Even though Java gives you the flexibility to provide a huge length for an identifier a very long name is impractical and difficult to manage.

## CH 4: VALUES AND DATA TYPES

1. What is Unicode?

Ans.Java uses the Unicode character set. It is a two-byte code set that has characters representing almost all characters in almost all human alphabets and writing systems around the world. Unicode is a superset of ASCII and the numbers 0 to 127 has the same meaning in both ASCII and Unicode formats.

1. State the differences between Syntax errors and Logical errors.

Ans: The compiler can only translate a program if the program is syntactically correct; otherwise the compilation fails and you will not be able to run your program. Syntax refers to the structure of your program and the rules about that structure.

The second type of error is a run-time error, so-called because the error does not appear until you run the program. In Java, run-time errors occur when the interpreter is running the byte code and something goes wrong.

1. Differentiate between Testing and Debugging.

Ans: Testing is the process of checking program logic manually to ensure whether it contains any error or not. Debugging is the process of removing errors from a program.

1. What are tokens?

Ans: Tokens are the smallest individual units in a program. Also known as lexical units. The types of tokens: keywords, identifiers, literals , punctuators and operators.

1. What are keywords? Can keywords be used as a identifiers?

Ans: Keywords are the reserved words that convey a special meaning to the language compiler. No, keywords are reserved for special purpose and must not be used as identifiers.

1. What is an identifier? What is the identifier formatting rule of Java? OR What are the rules for naming a variable?

Ans: Identifiers are names given by the programmer to different parts of a program e.g. variables, functions, classes,packages etc. The rules of naming identifiers in Java.

1. Can contain alphabets, digits, dollar sign and underscore.
2. Must not start with a digit.
3. Cannot be a Java keywords.
4. Can have any length and are case-sensitive.
5. State the difference between Token and Identifier.[2008]

Ans: The smallest individual unit of a program is known as Token.every unit that makes a statem is known as atoken. The following Tokens are available in Java: Keywords, Identifiers, Literals, Punctuations, Operators.

Identifiers are names given by the programmer to different parts of a program e.g. variables, functions, classes etc.

1. What are literals? How many types of integer literals are available in Java?

Ans: A literal is sequence of characters used in a program to represent a constant value. For example „A‟ is a literal that represents the value A of type char, and 17L is a literal that represents the number 17 as value of type long. Different types of literals available in Java, they are: Integer literal, Float literal, Boolean literal, Character literal, String literal and null literal.

Integer literals are of three types – Decimal Integer literal (9) , Octal integer literal(09) and Hexadecimal integer literal (0X9)

1. What is an integer constant? Write integer forming rule of Java.

Ans: Integer constants are whole numbers without any decimal part. The rule for forming an integer constants is: An integer constant must have at least one digit and cannot contain a decimal point. It may contains + or – sign. A number with no sign is interpreted to be positive.

1. What do you mean by Escape sequence and name few escape sequences in Java?

Ans: Java has certain nongraphic characters (nongraphic characters are those characters that cannot be typed directly from keyboard e.g. backspace, tab, carriage return etc.). Those nongraphic characters can be represented by escape sequence. An escape sequence is represented by backslash followed by one or more character. The few escape sequence characters are: \n for new line, \t for Horizontal Tab, \v for Vertical Tab,

\b for Backspace, \” for Double Quotes etc.

1. What is meant by a floating constant in Java? How many ways can a floating constant be represented into?

Ans: Floating constants are real numbers with fractional points. A floating constant can either be represented in fractional form (e.g. 0.00456) or in exponent form

(e.g. 4.56E-03)

1. Differentiate between Integer and Floating type constants.

Ans: Integer constants are the whole numbers (without decimal points). e.g. 1231. Floating point constants are fractional numbers (number with decimal points). e.g. 14.2356

1. Write the following real constants into fractional form: 0.113E04, 0.417E-04, 0.4E-05, 0.123E02

Ans: 0.113E04 becomes 1130, 0.417E-04 becomes .0000417, 0.4E-05 becomes .000004, 0.123E02

becomes 12.3

1. What is a type or „Data Type‟? How this term is related to programming?

Ans: A type or datatype represents a set of possible values. When we specify that a variable has certain type, we are saying what values the expression can have. For example to say that a variable is of type int says that integer values in a certain range can be stored in that variable.

Datatypes in programming are the means to identify the type of data and the associated operations of handling it.

1. What is primitive data type? Name its different types.

Ans: Primitive(fundamental) data types are those that are not composed of other data types. Numeric Integral, Fractional, character and boolean are different primitive data types.

1. Differentiate the two kinds of data types? [2006]

Ans: The two types of data types are: Primitive and non-primitive/composite/user define data types.

The primitive data types are the basic datatypes provided by the language. It is used to define and hold a value in a named variable. They are: byte, short, int, long, float, double, char and Boolean.

The non-primitive/reference data types are created using primitive datatypes. It is used to store the memory address of an object. The reference datatypes are: class, array and interface.

1. How many bytes occupied by the following data types: byte, short, int, long, float, double, char, boolean.

Ans: char-2 byte, byte-1 byte, short-2 bytes, int-4 bytes, long-8 bytes, float-4 bytes, double-8 bytes, boolean- Java reserve 8 bits but only use 1 bit.

1. What is the range of the following data types: byte, short, int, long, float, double, char, boolean.

Ans: byte -27 to +27-1

short -215 to 215-1

int -231 to 231-1

long -263 to 263-1

float -3.4×1038 to 3.4×1038

double -1.7×10308 to 1.7×10308

char 0 to 65536

boolean true or false

1. What do you mean by variables? [2006]

Ans: A variable is a named memory location, which holds a data value of a particular data types. E.g. double p;

1. What do you mean by variables? What do you mean by dynamic initialization of a variable?

Ans: A variable is a named memory location, which holds a data value of a particular data types. When a method or functions is called and the return value is initialized to a variable is called dynamic initialisation.

e.g. double p=Math.pow(2,3);

1. Differentiate between character and boolean data type.

Ans.

Character

1. stores Unicode character
2. Size 2 bytes
3. Range : 0 to 65,535 Boolean
4. used to represent a single true/false value
5. Java reserves 8 bits but uses 1 bit
6. Range : true or false
7. How can you change a variable to a constant? Give an example.

Ans. The keyword final makes a variable as constant , so that the its value cannot be changed during program execution.

e.g final double rate=5.5;

CH 5: OPERATORS AND EXPRESSIONS

1. What is the function of an operator? Name the types.

Ans: Operators are special symbols that represent operations that can be carried out on variables, constants or expressions. An operator is mostly an symbol and sometimes a keyword also.

The different types of operators are:

Arithmetic operator( unary operators, binary operators, increment/decrement operators) Relational operators

Logical Operators Assignment operators Conditional operator ?:

New operator Dot(.) operator

1. What are arithmetic operators?

Ans: Arithmetic operators are used for various mathematical calculations. The result of an arithmetical expression is numerical values. Arithmetical operators are of following types: Unary and Binary operators and increment/decrement operator.

1. Write major difference between the unary and binary operators?

Ans: The operators that acts on one operand are referred to as Unary Operator. There are two Unary operators Unary + operator and Unary – operator. The operators that acts upon two operands are referred to as Binary Operator. The Binary Operators are Addition (+), Subtraction (-

), Multiplication (\*), Division (/) and Modulus (%).

1. What is increment operator? What are postfix and prefix increment operators?

Ans: The „++‟ operator is called increment operator. The increment operators add 1 to its operand. These are two types (i) Prefix and (ii) Postfix The prefix version comes before the operand for e.g.

++a, where as postfix comes after the operand e.g. a++

1. Find the value of x after evaluating x += x++ + –x + 4 where x=3 before the evaluation. Explain your answer.

Ans: Result is 13, because x++ is 3, –x is 2 + 4 the answer is 9 add this with x that is 3 it becomes 12 and due to pre increment of x++ the result becomes 13.

1. What do you mean by Relational Operators?

Ans: Relational operators are used to determine the relationship between different operands. These are used in work of compression also. The relational expression (condition) returns 0 if the

relation is false and return 1 if the relation is true. < (less then), > (greater then), <= (less then equals to), >= (greater then equals to), == (equals to), != (not equals to).

1. What is Logical operators?

Ans: The logical operators combine the result of or more than two expressions. The mode of connecting relationship in these expressions refers as logical and the expressions are called logical expression. The logical expression returns 1 if the result is true otherwise it returns 0. The logical operators provided by Java are && Logical AND, || Logical OR, ! Logical NOT.

1. What do you mean by Assignment Statement or Assignment Operator?

Ans: Assignment operator is represent by symbol „=‟. It takes the value on the right and stores it. It assigns one value to the other.

1. Illustrate „?‟ operator with an example? [2009]

Ans: It is a conditional operator, that stores a value depending upon a condition. This operator is also known as ternary operator. The syntax for this operator is expression1?expression2:expression3 . and the example is bonus=sales>15000?250:50;

1. What is the purpose of new operator? [2006]

Ans: We can use new operator to create a new objects or new array. Ex. myCar obj = new myCar();

int arr[] = new int[5];

1. What do you mean by precedence? Illustrate with the help of example.

Ans: Precedence is the order in which a program evaluates the operations in a formula or expression. All operators have precedence value. An operator with higher precedence value is evaluated first then the operator having lower precedence value. consider the following example

x = 5 + 4 \*9;

The value of this expression is 29 not 54 or 34. Multiplication has been performed first in this expression.

1. What is operands?

Ans: An operator acts on different data items/entities called operands. The objects of the operation are referred as operands.

1. What do you mean by constant? How you declare a variable as constant variables.

Ans: The memory variables/locations whose values cannot be changed within the program is called constants. The keyword final makes a variable as constants.

e.g. final int x=203;

1. What do you mean by type conversion? What is the difference between implicit and explicit type conversion explain with example.

Ans: The process of converting one predefined type into another is called Type Conversion.

An implicit type conversion is a conversion performed by the compiler. The Java compiler converts all operands up to the type of the largest operand. This is also known as type promotion. e.g. „c‟-

32 is converted to int type. Where as an explicit type conversion is user defined that forces an expression to be of specific type, this also known as type casting. e.g. (float)(x+y/2)

1. Explain the methods print() and println()?

Ans: A computer program is written to manipulate a given set of data and to display or print the results. Java supports two output methods that can be used to send the results to the screen. print() method println() method.

The print() method sends information into a buffer. This buffer is not flushed until a new line (or end-of-line) character is sent. As a result print() method prints output on one line.

The println() method by contrast takes the information provided and displays it on a line followed by a line feed.

1. What is an Expression? Explain its different types.

Ans: An Expression is any statement which is composed of one or more operands and which returns a value. It may be combination of operators, variables and constants. There are different types of expressions.

1. Constant Expressions: 8 \* 12 /2
2. Integral Expressions: formed by connecting integer constants x = (a + b)/2
3. Logical Expressions: a > b or a!=b
4. Boolean expression: (y+z) >= (x/z)
5. Mention two different styles of expressing a comment in a program. [2005] Ans: The two ways of inserting a comments in a program are:
   1. using // single line comments
   2. using /\* \*/ multiple line comments
6. What is a compound Statement? Give an Example. [2005]

Ans: It is a block of code containing more then one executable statement. In Java the { } is called block and the statements written under {} is called compound statements or block statement. The {

} opening and closing sc aces indicates the start and end of a compound statement.

for(int i=1;i<=5;i++)

{

System.out.println(“Hello”); System.out.println(“How”); System.out.println(“are you?”);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[1](https://nilabja.wordpress.com/computers/theory-of-java/#comments)**What is Java?**  
Java is a 3rd Generation Object Oriented programming Language, developed by James Gosling in the year 1995. It was initially codenamed as Green and hypothetically called Oak named after the Oak tree outside Gosling’s house.  
It is case sensitive and has a unique two step translation process that includes both Compilation and Interpretation which makes Java Platform independent.  
**What is Procedure Oriented Programming Language?**  
Pop is a Procedural Oriented Programming Language. In this, the stress is laid on function rather than data. Data may be kept floating throughout the Programming. Hence by scanning the whole program from beginning to the end and we can rectify any error caused. E.g. of POP based language are GW-BASIC, C, etc…  
Characteristics of POP  
1) Emphasis is laid on functions (Logical steps).  
2) Functions share global data.  
3) Data values can keep floating from one function to another.  
4) Uses top down approach of programming.  
**Drawbacks of POP**  
1) As Data values are global to all the functions, you may require making necessary changes in the functions due to change in the data values.  
2) It is not suitable to solve complex problems in real situations.  
What is **Object Oriented Programming Language**?  
OOP is an Object Oriented Programming Language. It is a modular approach to programming in which equal stress is given to data as well as methods and it allows the data be applied within the stipulated program area. It also provides reusability features to develop productivity logic.  
**Features of OOP**  
1. It gives equal stress on data and function.  
2. It makes the program simple by dividing it into a number of objects.  
3. The objects can be used as a bridge to have data flow from one function to another.  
4. Data can be easily modified without any change in the function.  
Advantages of OOP  
1) We can extend the use of existing class through inheritance.  
2) Using the concept data hiding can generate secured program.  
3) We can create different modules in our project through objects.  
4) Multiple instances of an object can be generated to co-exist without any interference.  
5) It is highly beneficial to solve complex problems.  
6) It is easy to modified and maintain software complexity.  
Examples of OOP based languages are Simula, C++, Java, Python, and Smalltalk etc…  
**Basic Principles of OOP**  
1) Encapsulation – Binding up of data members and member functions together into a single unit called (class) is called Encapsulation.  
2) Data Abstraction – Act of representing only essential features without including its background details is called Data Abstraction.  
3) Polymorphism – The ability of a method to behave in more than one form is called polymorphism. Function Overloading is a way to implement it, in which more than one function has the same name but different argument list.  
4) Inheritance – The ability of a class to derive the properties from another class is called Inheritance. The class that inherits is called the Derived / Sub / Daughter class and the class from which it is derived is called Base / Super / Parent class. The keyword used to implement Inheritance is ‘extends’.  
5) Object – It is an identifiable entity with some characteristics and behaviour.  
6) Class – It is a blueprint that represents a set of objects which share common characteristics and behaviour.  
Related Terms :  
**Data binding**– is the process to link to the function call with function signature at run-time i.e., during execution of a program.  
**Attribute**:-The characteristics which make every object having its identity make them distinguished is called attribute.  
**Behaviour**– The behaviour of any class or object is represented through various functions and methods.  
**Message Passing** – When object need to interact with one another they do so by passing information to and from one another, this is called Message Passing  
**Data Hiding** – is the data which cannot be accessed directly outside, class premises although they are available in the same program  
**Garbage Collection** – The Garbage-collected Heap is where the objects in Java programs are stored. Whenever we allocate an object using new operator, the heap comes into picture and memory is allocated from there. Java does this automatically using Garbage collection mechanism, using an algorithm named Mark & Sweep logic. Remember that the local object reference resides on Stack but the actual object resides in Heap only. Also, arrays in Java are objects, hence they also resides in Garbage-collected Heap.  
**Different Types of Java Programs**  
1) **Standalone Application** – An application program that is developed by programmers which consists of various methods and these methods being invoked from within a ‘main ( )’ method is called standalone application.  
2) **Internet Applets**– These are small java programs that are embedded in a web page and which requires a web browser to be run/executed.  
3) **Servlets**– They extend the functionality of web browser.  
Source Code – The High level language code written by programmers which is not understandable by the computer is known as Source Code. It has an extension of \*.java  
Byte Code – is an intermediate code that consists of a set of pseudo machine language instructions that are understood by the JVM and are independent of the underlying hardware. It is called byte code because each chunk of code is of 8 BITS (1 byte = 8 Bits).  
The byte code files have an extension of \*.class  
**Object Code**– The interpreter of Java named JVM (which stands for Java Virtual Machine) then converts the intermediate Byte Code into machine specific executable which runs on the host machine.  
API (Application Programming Interface) – consists of libraries of pre compiled code that programmers can use in their application and programs for designing softwares. Hence we can say that Java API consists of functions and variables that programmers can reuse.  
**Java Packages** – Packages are collection of similar nature classes. A package can be created by using the keyword ‘package’ and the keyword used to include a package in our program is ‘import’. Java contains an extensive library of pre-written classes grouped together into packages –  
 θjava.lang.\* It is a default package containing String, Math, Integer, thread etc.  
 θjava.io.\* It is the basic Input Output package of Java  
 θjava.util.\* The java utility package  
 java.applet.\* The applet packageθ  
 θjava.net.\* The Networking package of Java  
 θjava.awt.\* It stands for Abstract Windows Toolkit, for designing Windows based applications.  
**Basic Features of Java:-**  
1) Write once run anywhere(WORA)  
2) Platform Independent  
3) Offers many security features to make its program safe and secure  
4) Built in graphics & supports multimedia  
5) Light Weight code  
6) Open Product… i.e., freely available to all.  
**Blue – J**  
Blue-J is an **Integrated Development Environment (IDE)** for writing, editing, compiling, testing, executing and debugging the Java programming language, developed mainly for educational purposes, but also suitable for small-scale software development. Blue J was initially designed at University of Kent by **David Barns** and **Michael Kollins**  
The main screen graphically shows the class structure of an application under development (in an UML-like diagram), and objects can be interactively created and tested. This interaction facility, combined with a clean, simple user interface, allows easy experimentation with objects under development.  
**Features of Blue – J**  
1. Simplicity of the interface – The GUI user friendly interface is simpler than in full-scale professional environments, and thus easier to learn.  
2. The “Code Pad” – The code pad is a tool that instantly evaluates arbitrary Java expressions and statements.  
3. Regression testing  
4. Provision for syntax highlighting  
5. Built in debugger.  
6. Program input/output i.e. In Blue-J both the parameter values for and the return values from method calls can be entered / inspected directly.  
7. Java ME support – Java ME (Micro Edition) projects can be developed and deployed from Blue-J.  
**IDE for Java (other than Blue J)**  
Sl IDE Brief Description  
1 Eclipse Free, open source IDE. The most popular of the Java IDEs, but harder to set up and configure than the commercial ones.  
2 J Builder Lots of J2EE support including UML and app server integration.  
3 Emacs This is a powerful, free editor that does color syntax highlighting, automatic indentation, and parenthesis/brace balancing.  
4 M7 Nitro X Expensive but very powerful J2EE IDE based on Eclipse.  
**Java Tokens & Character Set**  
It is a set of valid characters that a language can recognize. The character set of Java is UNICODE. This has been selected due its vastness of characters. It is 2 byte character code. The first 128 characters of Unicode are identical to the ASCII character set.  
Tokens are smallest individual units of a Java program. There are 5 different types of tokens in Java –  
1. Keywords  
2. Identifiers  
3. Literals  
4. Operators  
5. Separators Hint: Try to remember KILOS  
**Keywords**are the reserved words which convey a special meaning to a language compiler. They can not be used for any other purpose like function name, variable name or object names.  
There are around 48 keywords, of which const and goto have been reserved even though they are not currently being used for any purpose true, false and null might appear to be keywords but actually they are literals, and hence are reserved words.  
**Identifiers**are fundamental building blocks of a program. They are named memory locations or simply names given to variables, objects, classes, functions, arrays etc… They are case sensitive.  
There are certain Identifier naming rules –  
1. They can consist of infinite number of alphabets and digits  
2. They must start with an alphabet and not with a digit  
3. They can not be a keyword or reserved word  
4. They cannot have any spaces  
5. They cannot contain any special character other than \_(underscore), $(dollar)  
While naming identifiers there are certain conventions that make them identifiable  
1. Class names begins with an Uppercase character  
2. Identifiers having multiple words, the first character of 2nd and subsequent words are in Uppercase  
3. Constants are in all CAPITALS  
Comments (Remarks):- refers to statements which are ignored by the java compiler and increase the readability of a program. These are used to state a purpose of the instructions used in a program.  
**Literal**are the fixed values that are assigned to variables which do not change its values during program execution. They are of 5 types –  
**1.Integer Literal**  
i.Decimal  
ii.Octal  
iii.Hexadecimal  
To represent Decimal Integers we simply use the integer type number as literal, e.g. int p = 45; So, System.out.println(p); // Shows 45 as output  
To represent Octal Integers however we precede the number with a 0 (zero) as literal, e.g. int p = 045; System.out.println (p); // Shows 37 as output coz, (45) in octal is 37 in decimal number system.  
Moreover to represent any Hexadecimal Integer we use 0x (Zero and lowercase letter x) preceding the number as hexadecimal literal e.g. int p = 0x45;  
System.out.println (p); // Shows 69 as output coz, (45) in hexadecimal is 69 in decimal number system.  
2. **Real Literals**  
i. Floating point literals have 7 digits of precision after the decimal point  
ii. Double literals have 16 digits of precision after decimal point  
3. **Character Literals** are enclosed within a pair of single quotes  
char ch = ‘A’;  
Character type user inputs are taken in the following way –  
e.g. char ch = (char) in.read ( ) ;  
4. **Boolean Literal**  
There are only 2 types of boolean literals i.e. true & false the default being false.  
5. **Null Literal**  
Operator – are the entities or symbols that tell the compiler that what operation has to be performed with one, two or three number of operands within a given expression.  
There are approximately a set of 37 different operators that Java uses  
Operators can either operate on 1, 2 or 3 operands and accordingly named as –  
 Unary (works on 1 operand) e.g. + + and – –  
 Binary (works on 2 operands) e.g. +, – , \* , / , >, , = , > , << )  
 Special Operators (this , dot (.) , new, (type) , instanceof )θ  
**Data types** – are the entities that tell the compiler that which variable will hold what kind of values.  
Primitive data types – Also called as Basic Data type. They are pre defined or built in data types because they are already built in java. E.g. byte, short, int, long, float, double, char, boolean.  
Non Primitive Data Types: Directly or indirectly dependent on primitive data types. They store the data values with reference to the address or locations. So, they are called as Reference Data Types. E.g. array, classes interfaces etc…  
**Range, Size and hierarchy of various primitive data types**  
Type ….Size…. Range  
boolean 1 byte (uses 1 bit) true and false  
byte 8 bits (1 bytes) – 128 to + 127  
char 16 bits (2 bytes) 0 to 65,535  
short 16 bits (2 bytes) – 32768 to +32767  
int 32 bits (4bytes) – 2 31 to + 231 – 1  
long 64 bits (8bytes) – 2 63 to + 263 – 1  
float 32 bits (4bytes) – 3.4E+38 to +3.4E+38  
double 64 bits (8bytes) – 1.7E +308 to 1.7E+308  
Type conversion The process of convert one primitive datatype to another either from lower precision to higher or vice versa I known as type conversion. This can be achieved in two ways –  
**Implicit Data Type Conversion** – Such Type of conversion is performed by the compiler without programmer’s intervention. In this type of conversion a datatype of lower precision gets converted to a datatype with higher precision.  
e.g. int a = 10;  
double d = a;  
System.out.println(d);  
The above code snippet gives 10.0 as output; this clearly indicates that the int datatype got automatically converted to double.  
**Explicit data type Conversion** – Such type of conversion is user defined conversion which forces an expression to be converted into specific type. Generally this is done to force a conversion from higher precision data type to lower.  
e.g. double pi = 3.1415;  
int i = pi;  
The above statement is considered illegal and hence gives an ERROR message “POSSIBLE LOSS OF PRECISION”  
Hence the above code should be Type Casted in the following manner –  
int i = (int) pi;  
System.out.println(i);  
Now, the output to the above code will be 3  
**Potential problems with Type Casting**  
1. When a datatype is converted from double to float results in loss of precision i.e. a double datatype has 16 digits after decimal point which gets truncated to 7 digits.  
2. When a datatype is converted from double or float to integer type results in loss of entire fractional part leaving behind only the integral part.  
3. When a higher order integer datatype like int or long is converted to byte or short might result in the data going out of range.  
Take an example  
int a = 130;  
byte b = (byte) a;  
System.out.println(b);  
The output for the above code is – 126  
Explanation for the above –  
The above code tries to type cast int to byte, the data range for int is – 2 31 to + 231 – 1 whereas that of byte is – 128 to + 127, hence think in terms of a Number Line, when we start filling 130 in the number line starting from 0 (zero) we reach the end of the line at 127 and so the filling process continues from the –ve end and finally reaches – 126, hence the output…  
**Operators in Java**  
**Operator**– are the entities or symbols that tell the compiler that what operation has to be performed with one, two or three number of operands within a given expression.  
There are approximately a set of **37** different operators that Java uses  
Operators can either operate on 1, 2 or 3 operands and accordingly named as –

1. **Unary** (works on 1 operand) e.g. + + and – –
2. **Binary** (works on 2 operands) e.g. +, – , \* , / , >, <, == etc…
3. **Ternary** (works on 3 operands) e.g. ? :

**Operators can categorized as –**

1. Arithmetic Operators (+ , – , \* , / , %)
2. Relational Operators (> , < , >= , <= , == , !=)
3. Logical Operators (&&, || , !)
4. Conditional Operators (? : )
5. Bitwise Operators (&, | , ~ , ^)
6. Shift Operators (>> , << )
7. Special Operators (this , dot (.) ,  new, (type) , instanceOf )

**Shortcut Operations in Java**  
Java implements shortcut methods to work with basic arithmetic operators like  
+, – , \* , / and %  
Hence, a = a + 5; can be written as a + = 5;  
And the above + = operator is called Addition Assignment operator.  
Similarly we have –

* Subtraction Assignment – =
* Multiplication Assignment \*=
* Division Assignment /=
* Modulus Assignment %=

**Unary Operators**  
*Post/Pre – Increment and Post/Pre – Decrement Operators*  
a = a + 1 can also be written as either a++ (**Post Increment**) or ++a (**Pre Increment**)  
a = a – 1 can also be written as either a– – (**Post Decrement**) or – –a (**Pre Decrement**)  
**The difference between Pre and Post is** **–**  
In a **Postfix** operator like Post Increment or Post Decrement the value of the operand is first used and then increased or decreased as the case may be for e.g.  
int a = 5;  
System.out.println(a++);  
Gives 5 as output as the value of ‘a’ is first used and then increased to 6, so if we have one more print statement after the above stated line will give 6 as output.  
System.out.println(a);          // Output is 6  
In a **Prefix** operator like Pre Increment or Pre Decrement the value of the operand is first increased or decreased as the case may be and then used for e.g.  
int a = 5;  
System.out.println(++a);  
Gives 6 as output as the value of ‘a’ is first incremented and then displayed.  
**Relational Operators**are used to find the relationship between two quantities, whether they are equal, un-equal, greater than less than etc…  
**Symbol**  
**Relationship**  
  
>  
Greater than  
  
<  
Less than  
  
>=  
Greater than or equal to  
  
<=  
Less than or equal to  
  
==  
Equal to  
  
!=  
Not equal to  
**Logical Operators**are used to join 2 or more relational operators and of 3 types –

1. && (and) returns true when both or all the relations are true
2. || (or) returns true when any one of the relations are true
3. ! (not) inverts true to false and false to true

Say if the value of a=4, b=6 then…  
if(a>5 && b>5) will return false coz both the conditions are not true  
if(a>5 || b>5) will return true coz one of the conditions is true  
if(!true) will return false  
It is not mandatory to have a relation always inside an ‘if’, even without any ‘if’ clause we can check the return of any relation like –  
Say the value of x=7; then  
System.out.println(x==7); will give the output as true, and if we place (!) in front of the relation, System.out.println( ! (x==7)); will invert the output from true to false.  
**Bitwise and Shift Operators**  
**Operator**  
**Name**  
**Description**  
**Example**  
**Result**  
  
a & b  
and  
1 if both bits are 1 and 0 in all other cases  
3 & 5  
1  
  
a | b  
or  
0 if both the bits are 0, rest all other cases 1  
3 | 5  
7  
  
a ^ b  
xor  
1 if two bits are different and 0 in all other cases  
3 ^ 5  
6  
  
~a  
not  
Inverts the bits, after transforming the entire number into full bit representation  
~3  
-4  
  
n << p  
left shift  
Shifts the bits of n towards left by p positions thereby adding Zeros to the extreme LHS  
3 << 2  
12  
  
n >> p  
right shift  
Shifts the bits of n towards right by p positions resulting in loss of bits from LHS  
5 >> 2  
1  
  
**In all the above examples the operands (values) must be converted to Binary and then implemented with respective operators except for Right and Left Shift operators where only the variable ‘n’ is converted to binary and not ‘p’**  
Don’t confuse **&&**, which is the ***logical****and*, with **&**, which is the uncommon ***bitwise****and*. Although the bitwise *and*can also be used with boolean operands, this is extremely rare and is almost always a programming error.  
**Conditional Operator**  
The only Ternary operator of Java is the Conditional Operator, **? :** which can be used as an alternative to if…else  
Syntax: variable = condition ? true : false ;  
e.g. int n = 100>50 ? 1 : 0;  
In the above code the value of ‘n’ becomes 1 as the condition 100>50 is true, had it been wrong the value of ‘n’ would have been 0;  
**Special Operators**  
q     **[ ]** this operator is used to assign the size for an array and to access any particular element within an array.  
q     **( )** the parenthesis operator is used to enclose the arguments to a function in the function prototype line and invoke a function by enclosing the list of arguments.  
q     **Dot (.)** the dot operator is used to access the instance/class members of a class through an object or class name.  
q     **new** – the new operator is used to assign and allocate memory to new objects and arrays.  
q     **instanceof** – this operator checks whether the first argument is an instance of the second argument or not.  
e.g. **str** instanceof **String**                  // will return true if **str** is a **String** variable  
  
  
**Precedence Table**  
  
Operator Precedence  
  The entire list 1.      ( ) . [] (args) post ++ -- 2.      ! ~ unary + - pre ++ -- 3.      (type) new 4.      \*/ % 5.      + - 6.      << >> >>> 7.      < <= > >= instanceof 8.      == != 9.      & 10.  ^ 11.  | 12.  && 13.  || 14.  ?: 15.  = += -= etc  Try to Remember only these

1. ( )
2. unary operators
3. \* / %
4. + –
5. comparisons
6. && ||
7. ? :
8. = assignments

**Math class functions in Java** (The **Math** class belongs to the default package of Java named **java.lang**)  
**SlFunction Name**  
**Description**  
  
1  
Math.sin(r)Returns sine of angle ‘r’ given in radians  
2  
Math.cos(r)Returns cosine of angle ‘r’ given in radians  
3  
Math.tan(r)Returns tangent of angle ‘r’ given in radians  
4  
Math.asin(x)Returns the angle in radians whose sine value is ‘x’  
5  
Math.acos(x)Returns the angle in radians whose cosine value is ‘x’  
6  
Math.atan(x)Returns the angle in radians whose tangent value is ‘x’  
7  
Math.toDegrees(r)Converts Radians to Degrees  
8  
Math.toRadians(d)Converts Degrees to Radians  
9  
Math.min(a, b)Returns the smaller among a and b  
10  
Math.max(a, b)Returns the larger among a and b  
11  
Math.abs(a)Convert -ve magnitude to +ve magnitude.  
12  
Math.sqrt(x)Returns the square root of ‘x’  
13  
Math.pow(x, y)Returns x to the power of y  
14  
Math.exp(x)Returns ‘e’ to the power of ‘x’ where e=2.718 called Euler’s  Constant just like pi  
15  
Math.cbrt(x)Returns the cube root of ‘x’  
16  
Math.log(x)Returns the Natural logarithm of ‘x’  
17  
Math.floor(x)Returns the integral value of x less than or equal to ‘x’ but datatype returned is double, hence Math.floor(7.9) will give 7.0  
18  
Math.ceil(x)Returns the integral value of x greater than or equal to ‘x’ but datatype returned is double, hence Math.ceil(7.9) will give 8.019  
Math.rint(x)Returns the integral value of x by rounding off the fractional part to the nearest integer , but data type returned is double, hence Math.rint(7.9) will give 8.0  
20  
Math.round(x)Returns the integral value of x by rounding off the fractional part to the nearest integer, but datatype returned is int, hence Math.round(7.9) will give 8  
21  
Math.random()Generates a double type random number between 0 to 1  
22  
Math.PIThis returns the 16 digits precise value of PI(Note that it is not a function)**Comment Lines in Java**  
 // comments — single line commentθ  
After the two // characters, Java ignores everything to the end of the line. This is the most common type of comment.  
 /\* … \*/ comments — multiple line commentθ  
After the /\* characters, Java will ignore everything until it finds a \*/. This kind of comment can cross many lines, and is commonly used to “comment out” sections of code — making Java code into a comment while debugging a program.  
Always write braces. It is good programming style to always write the curly braces, {}, although they are not needed if the clause contains only a single statement. There are two reasons this is good.  
• **Reliability**. When code is modified, the indentation is such a strong indicator of structure that the programmer may not notice that the addition of a statement at the “correct” indentation level really isn’t included in the scope of the if statement. This is a surprisingly common error.  
• **Readability**. It is faster to read code with the braces because the reader doesn’t have to keep in mind whether they are dealing with an un-braced single statement or a braced block.  
Whitespace (e.g., a blank line, spaces)  
Insert blank lines to separate sections of your program. It’s like starting a new paragraph in English. The compiler ignores them — it’s for us humans.  
**Exception Handling in Java**  
Exception refers to any contradictory or unusual situation which can be encountered during a program execution. In Java Exceptions (or Errors) has been categorized broadly under three heading –  
1. **Syntax Errors** – A Syntax error occurs when the programmer writes a grammatically incorrect code that does not support the rules of programming of that particular language. This type of error gets detected by the compiler. E.g. missing semicolon, using keywords as identifier name, unmatched pair of { }, undefined functions, undeclared and un-initialized variables etc…  
2. **Logical Errors** – A Logical error is born out of a programmer who misunderstands the logic of the program and writes a syntactically correct code but which does not comply to the algorithm of the problem. E.g. The program demanded squaring of an integer type number but the programmer writes a code that determines the square root of the number.  
3. **Runtime Errors** – A Runtime error is shown when a syntactically and logically correct program with a set of inputs which is invalid for a given expression. The same expression is valid for another set of values. E.g. int quo = div / n, the expression is valid for all sets of positive values of n but not if n = 0.  
**Exception handling is ideal for –**  
1. Processing exceptional situations.  
2. Processing exceptions for components which cannot handle them directly.  
3. Processing exceptions for widely used components that should not process their own exceptions.  
4. Large Projects that requires uniform error processing.  
**Advantages** of exception handling:-  
1. It separates error handling code from the normal code.  
2. It enhances the readability.  
3. It makes clear and strong program code.  
The process of exception handling is implemented using a try … catch block, in which the try block contains the code which has probability of having errors during runtime and the catch block contains code to handle the situation.  
**Common Errors**  
0 < x < 100  
Comparison operators can be used with two numbers. Although you can write 0 < x = or <= instead of ==. For example, because the decimal number 0.1 can not be represented exactly in binary, (0.1 + 0.1 + 0.1) is not equal to 0.3  
String issues using = = and equals ( )  
Usually you want to know if the two Strings have the same value, for e.g.  
if (name == “Californication”) // Legal, but SURELY WRONG  
Reason is the = = operator is not comparing the values of name and Californication instead it is just comparing the references i.e. it is checking whether both of them are strings or not and hence returns true and to the programmer it reveals to be perfectly LEGAL. To prove this we can take another e.g. say, –  
String name = “Californication”;  
System.out.println (name. substring (0,3)); // Gives the output as Cal  
But when they are compared using = = gives the output as Un-Equal  
if (name. substring (0,3) = = “Cal”)  
System.out.println (“Equal”);  
else  
System.out.println (“Un-Equal”);  
Hence always use equals ( ) method of String class to compare the values of String class objects.  
**NaN**  
No exceptions are generated by floating-point operations. Instead of an interruption in execution, the result of an operation may be positive infinity, negative infinity, or NaN (not a number). Division by zero or overflow produces infinity. Subtracting two infinities produces a NaN.  
**Infinity**  
Similarly no exceptions are generated for statements like Math.pow(0,–1) and such statements will result in Infinity as output