**What is the difference between JDK and JRE?**

|  |
| --- |
| **1. JDK (Java Development Kit)**   * **Purpose:** Used for developing, compiling, and running Java applications. * **Includes:**   + JRE (Java Runtime Environment)   + Java Compiler (javac)   + Development tools (debugger, Javadoc, etc.) * **Who Needs It?** Developers who write and compile Java programs.   **2. JRE (Java Runtime Environment)**   * **Purpose:** Used only for running Java applications. * **Includes:**   + JVM (Java Virtual Machine)   + Java class libraries * **Who Needs It?** End-users who just want to run Java applications but don’t need to develop them. |

**What jdk(Java development kit) contains?**

|  |
| --- |
| It contains set of tools and JRE |

**What is JRE (Java Runtime Environment)?**

|  |
| --- |
| It contains JVM + JCL |

**What is JVM?**

|  |
| --- |
| Java Virtual Machine which is used to run the java programs |

**What is JCL (Java Class Library)**

|  |
| --- |
| Set of pre-defined classes. |

**What is a package?**

|  |
| --- |
| 1. A package is a container of classes. 2. A package contains collection classes. 3. A package is a folder which contains classes, interfaces, Enums and annotations |

**How many types of packages are there?**

|  |
| --- |
| There are two types of packages are there   * + 1. Pre- defined packages     2. User defined packages |

**What is java API?**

|  |
| --- |
| The API is derived as Application programming interface.  It is a collection of pre - defined packages. |

**List some pre-defined packages in java?**

|  |
| --- |
| 1. Java.lang 2. Java.util 3. Java.io 4. Java.awt 5. Java.text 6. Java.awt.event 7. Java.sql 8. Java.n et 9. Java.util.stream |

**In how many ways we write comments?**

|  |
| --- |
| there are three ways to write a comments   1. Single line comment(//…..) 2. Multi line comment(/\* ……. \*/) 3. Doucumentation comment (/\*\* ….. \*\*) |

**What is the use of import statement?**

|  |
| --- |
| It is a keyword and a statement which is used to import packages |

**What is JCL?**

|  |
| --- |
| Java class library which has a pre-defined class. |

**What is a keyword?**

|  |
| --- |
| A pre – defined word ,provided by java inventors . |

**What is a class?**

|  |
| --- |
| Class is a blue print of object. class is a block which contains fields and methods. |

**What is the use of “import java.lang.\*;” ?**

|  |
| --- |
| It is used to use the pre-defined class, annotations, Enums, interfaces existed in that packages. |

**Why we import classes of a package?**

|  |
| --- |
| To use those classes, you have to import them |

**What is the command used to compile the package programs?**

|  |
| --- |
| Javac -d . filename.java |

**How to run the program if main method class is existed in p1 package?**

|  |
| --- |
| java p1.Ayyo |

**Can we write print statement outside the function in Java?**

|  |
| --- |
| No we can’t, but we can write it in any block which is written in a class. |

**If you write print statement outside the function what error you will get?**

|  |
| --- |
|  |

**Can I declare a variable with in a class and outside the function?**

|  |
| --- |
| **Yes we can declare 2 types of variables**   * Instance variables (it is not declared as static) * Static variables (it is declared as static) |

**Can I write statements other than declarations in a class and outside the functions?**

|  |
| --- |
| No we can write only variable declaration(definition) statements. |

**If I write the “c=a\*s;” statement outside the function within a class what kind of error you will get?**

|  |
| --- |
|  |

**What is String, System which are used in our program?**

|  |
| --- |
| * These are a pre-defined classes existed in java.lang package |

**Can we use String class without importing java.lang package?**

|  |
| --- |
| Yes, we can because java.lang package is the default package, which means the compiler imports it automatically. |

**When the java.lang package will be imported automatically?**

|  |
| --- |
| During compilation |

**What is a string?**

|  |
| --- |
| A string is nothing but collection of characters represented by pair of double quotations .String is also a pre defined class which is existed in java.lang package  Ex-1: “ravi teja” -> string literal(value)  Ex-2: “Vamsi” -> string literal  Ex-3: “12345” -> string literal  Ex-4: “+-\*&^@” -> string literal |

**What is a concatenation symbol?**

|  |
| --- |
| If you find a ‘+’ symbol after or before a string literal it is called as a concatenation symbol |

**What is the use of concatenation symbol?**

|  |
| --- |
| It appends any value to the string  Ex: “a=”+10 result is “a=10” |

**What are the differences between print and println()?**

|  |  |
| --- | --- |
| Print | Println |
| 1. It prints the data without moving cursor to the new line | 1. It moves the cursor to the new line after printing the data. |
| 1. We can pass only one argument to the print function | 1. We can pass zero or one argument to the println function |

**What is a variable?**

|  |
| --- |
| 1. It a container to store a value 2. Variable is changeable |

**How to declare a variable?**

|  |
| --- |
| In Java we can declare a variable by using a data type  Syntax: <data-type> <var-name>[=value]; Ex: int a; Ex: float f=10.00f;  Syntax: <data-type> <var-1[=value]>[,var-2[=value],var-3[=value]…var-n[=value]]; |

**What is a data type?**

|  |
| --- |
| Def-1: The type of data we store in a variable is called as data type.  Def-2: it is a keyword or a class name or a combination of data type and one or more subscripts which is used to declare a variable? |

**Why we declare a variable?**

|  |
| --- |
| To allocate memory in RAM. |

**What are the types of variables?**

|  |
| --- |
| Based on the data we store variables are divided into 2 categories   1. Value type variables 2. Reference or object data types |

**What are value type variables?**

|  |
| --- |
| 1. in these variables we can store the value not address 2. to declare value type variables, we have to use primitive data types |

**What are reference variables?**

|  |
| --- |
| 1. In these variables we can store either address or null. 2. We can declare reference variables, by using class name, interface name, annotation name, Enum name or combination of data type and one or more subscripts. |

**What is a local variable?**

|  |
| --- |
| 1. It is a variable which is declared within a block existed in a class is called as local variable 2. Local variables are created either in java stack memory or native method stacks memory. 3. Local variables will not be initialized with default values, so you have to initialize them before usage. |

**Why java is called as extensible programming language?**

|  |
| --- |
| Because in Java we can use native methods |

**What are native methods?**

|  |
| --- |
| 1. Methods which are written in other languages like C are called as native methods 2. It is possible to use native methods in Java   Ex: public static native long currentTimeMillis(); method of System class is a native method |

**What is the first argument we should pass to the formatting method?**

|  |
| --- |
| String literal |

**How many arguments we can pass to the formatting method?**

|  |
| --- |
| 1 or more arguments |

**What are the formatting methods?**

|  |
| --- |
| Those are:   * 1. Printf()   2. Format() |

**What is the difference between printf() and format() method?**

|  |  |
| --- | --- |
| Actually **printf()** method is internally calls the **format()** method, that is actual code is existed in the format method. The printf() method is just provided for our convenience. |  |

**In which class printf() existed?**

|  |
| --- |
| It is existed in the PrintStream class; it takes 1 or more arguments. |

**What is a function?**

|  |
| --- |
| * A function is a block which contains re-usable set of statements * We write a function to perform a task * Once the function is written, to execute the statements of it, we have to call it |

**What is a static method?**

|  |
| --- |
| * 1. If a method is declared by using static keyword then it is called as static method   2. We can call a static method in another static method of same class directly.   Note: we can’t call non-static method (instance method) in a static method directly. Even though it is existed in the same class. |

**Who calls the main function?**

|  |
| --- |
| JVM calls the main function |

**What is the advantage of functions?**

|  |
| --- |
| Functions provide several advantages in programming:   1. **Code Reusability** – Functions allow you to reuse code multiple times without rewriting it, reducing redundancy. 2. **Modularity** – They break a large program into smaller, manageable parts, making the code easier to understand and maintain. 3. **Improved Readability** – Well-named functions make the code more readable and self-explanatory. 4. **Easier Debugging** – Errors are easier to find and fix when the code is modular, as you can test individual functions separately. 5. **Scalability** – Functions help in building scalable applications by organizing code efficiently. 6. **Encapsulation** – They allow you to hide implementation details and expose only necessary functionality, improving security and organization. 7. **Avoiding Code Duplication** – Instead of writing the same logic multiple times, you can call a function whenever needed. 8. **Efficient Memory Use** – Functions help optimize memory usage by executing only when called and releasing resources when done. |

**What is a parameter?**

|  |
| --- |
| * + 1. Variables which are declared within parenthesis of a function are called as parameters.     2. Parameters are separated by comma (,)     3. After last parameter no need to give semicolon or comma |

**What is a method?**

|  |
| --- |
| A method is a function written in a class. |

**What is the starting point of the program?**

|  |
| --- |
| 1. Main method 2. In Java main method should be a static method 3. In Java main method return type should be only void.   Note: In Java you should declare the main method as public and static |

**How to read total line at a time from keyboard?**

|  |
| --- |
| readLine() method of DataInputStream class  (or)  readLine() method of BufferedReader class |

**How can u read data from keyboard by using readLine() method of DataInputStream class?**

|  |
| --- |
| * readLine() method is a non-static method existed in DataInputStream class, so we have to call it by using object of DataInputStream class * readLine() method returns string only.. |

**What is an operator?**

|  |
| --- |
| Operator is a symbol, which is used to perform an operation. |

**What are different types of operators supported by Java?**

|  |
| --- |
| Ex: +, -, \*, /, %, (arithmetic operators)  Ex: <,>,<=,>=,==,!=,instanceof (relational operators  Ex: &&,||, ! (logical operators)  Ex: &,|,^,<<,>>, ~, >>>(Bitwise Operators)(zero fill right shift operator)  Ex: = (assignment operator)  Ex: new (memory allocation operator), . (dot is called as memory access operator), () type cast operator  Ex: ++,--, +=,-=,\*=,/=,%= (short cut operators) |

**What is the drawback of readLine() of DataInputStream class and why it is deprecated?**

|  |
| --- |
| * by using this method, we can read only the characters which are in a range of 0 to 255 (ASCII character set). * this method can’t read other than ASCII character set (i.e. it can’t read all the characters of Unicode character set). * The method has been deprecated since Java 1.1 because it does not properly convert bytes to characters in a way that supports internationalization. * readLine() reads input as a **byte stream** and does not properly decode characters according to the character encoding (e.g., UTF-8). This can lead to **incorrect character conversions**, especially for non-ASCII characters. |

**What is the readLine() method of BufferedReader class**

|  |
| --- |
| * 1. by using this method, we can read the characters which are in Unicode character set. |

**Where the parseInt() method is existed?**

|  |
| --- |
| * It is existed in Integer class * It takes string as an argument and converts it into int value and returns that int value |

**What is Relational Operators?**

|  |
| --- |
| Relational Operators are used to compare 2 values(operands), after comparison these operators gives us a Boolean value(true/false) |

**What is an operand?**

|  |
| --- |
| It is a value on which the operator performs an operation.  Ex: 10+20 Here 10,20 are operands and + is an operator |

**What is Logical Operators (Boolean operators)?**

|  |
| --- |
| * 1. Logical operators are used to perform operation on Boolean values only   2. These operators gives us Boolean values as a result. * Logical and operator (&&): it gives us true if both operands are true, otherwise false * Logical or operator (||):it gives us true if any one or both operands are true, otherwise false * Logical not operator (!): it returns true if operands is false vice versa. |

**How parseBoolean() method works?**

|  |
| --- |
| * 1. It converts “TRUE” to true and returns it   2. It converts “true” to true and returns it   3. If any string other than “true” (case-insensitive) is passed then It returns false |

**How to read a single character (within a range of 0 to 255)?**

|  |
| --- |
| By using System.in.read() |

**Can we declare a local variable as static?**

|  |
| --- |
| No, local variables can’t be either static or instance |

**What is a literal?**

|  |
| --- |
| 1. It is a value used in a program |

**What are different types of literals supported by Java?**

|  |
| --- |
| 1. Integer literal (10,20) 2. Float literal (10.0,2.0) 3. String literal(“madhu”,”123”) 4. Boolean literal(true,false) 5. Character literal (‘a’,’1’) |

**What is a token?**

|  |
| --- |
| Each and every individual unit in a program is called as token. |

**list of tokens supported by Java?**

|  |
| --- |
| 1. Keywords 2. Identifiers 3. Literals 4. Operators 5. Separators 6. Special characters 7. Comments |

**What is a comment?**

|  |
| --- |
| A **comment** in Java is a piece of text in the code that is ignored by the compiler and is used to provide explanations, notes, or documentation for better code readability. Java supports three types of comments:  **1. Single-line Comment (//)**   * Used for short, single-line explanations. * Anything after // on the same line is ignored by the compiler.   **2. Multi-line Comment (/\* ... \*/)**   * Used for longer explanations spanning multiple lines.   **3. Javadoc Comment (/\*\* ... \*/)**   * Used for generating documentation using the **Javadoc** tool. * Typically used for describing classes, methods, and parameters. |

**What is String in “String args[]”?**

|  |
| --- |
| It is a pre-defined class existed in java.lang package |

**What is a System in “System.out.println()”?**

|  |
| --- |
| It is a pre-defined class existed in java.lang package |

**What is a println() in “System.out.println()”?**

|  |
| --- |
| It is a pre-defined method |

**What is a reference variable?**

|  |
| --- |
| If we declare a variable by using class name as a data type. Then those variables are called as reference variables (like pointer variables in C/C++). |

**What is a reference data type?**

|  |
| --- |
| It is a data type which is used to declare reference variables. |

**What are conditional statements in java?**

|  |
| --- |
| Conditional statements allow the program to make decisions based on certain conditions. The main conditional statements available in Java are:   * + 1. If statement     2. If…else… statement     3. Else…if. Ladder (if..else..if) (if..else if…else)     4. Nested if statement     5. Switch statement     6. Ternary operator     7. While     8. for     9. Do..while     10. Foreach |

**What is if - statement?**

|  |
| --- |
| If you want to execute one or more statements whenever a condition is true then we write if statement. |

**What is the syntax of IF – statement?**

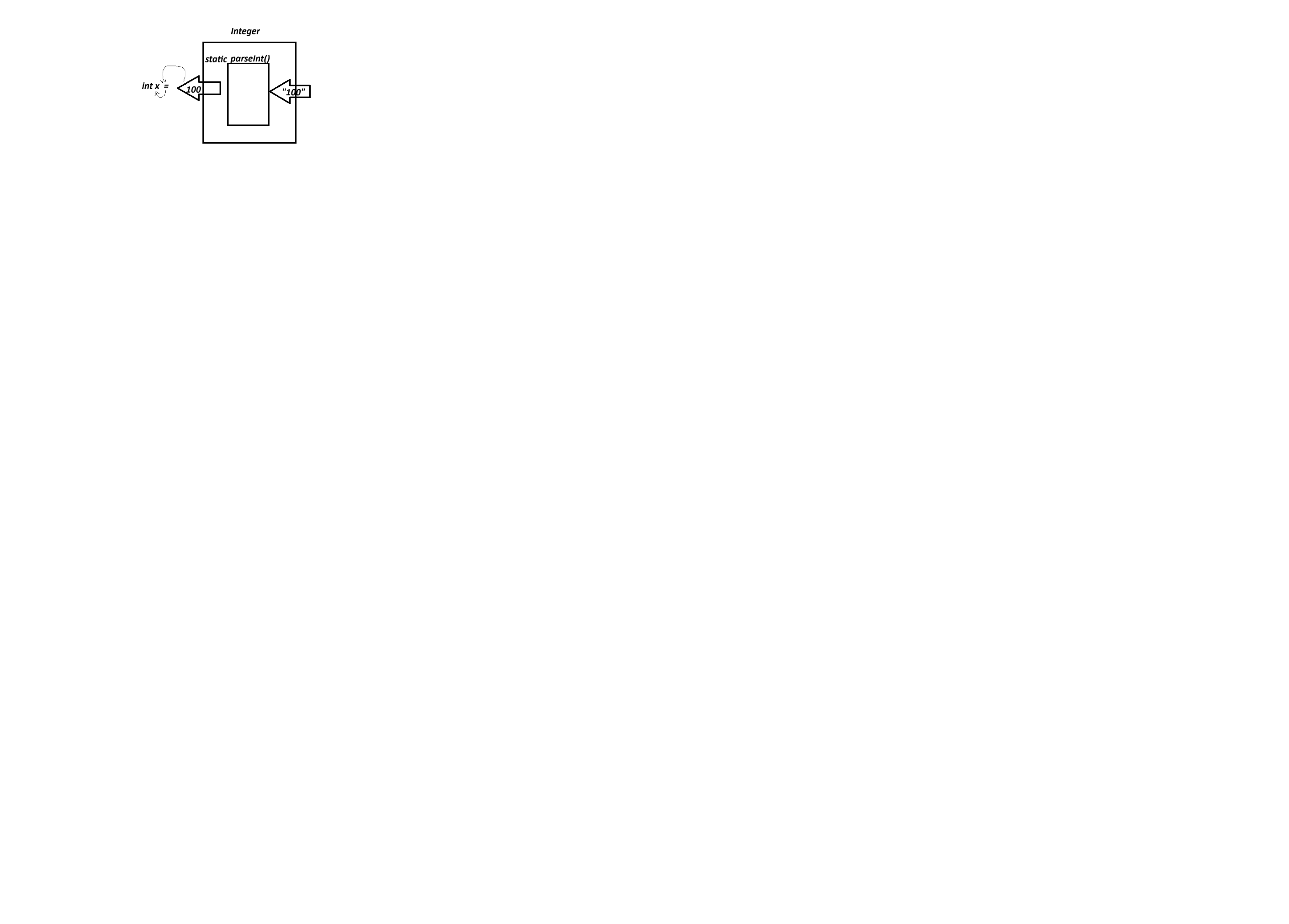
|  |
| --- |
| Syntax-1: if (condition)  Single-statement-1;  Syntax-2: if (condition){  Stmt-1;  Stmt-2;  …………  Stmt-n;  } |

**What is Integer in Integer.parseInt() statement?**

|  |
| --- |
| It is a pre-defined class existed in java.lang package |

**What is parseInt() in Integer.parseInt() statement?**

|  |
| --- |
| * It is a pre-defined method existed in Integer class * This method takes string as an argument and returns int(primitive int) value. |



**How we can call the static method?**

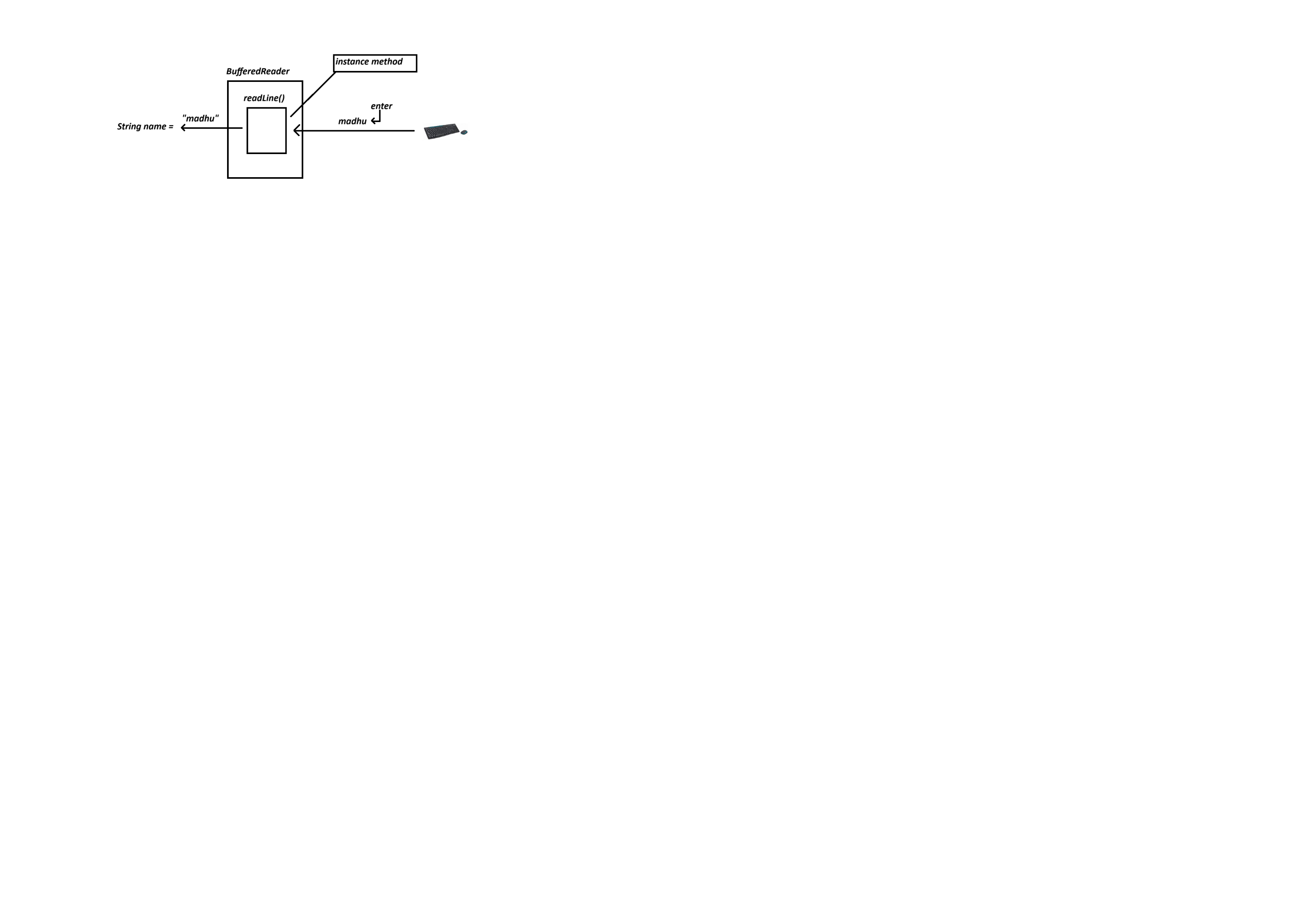
|  |
| --- |
| We can call the static method by using class name or a reference variable name(don’t bother whether it contains address of the object or null)  Ex: int x=Integer.parseInt(“100:); |

**Why we write a class?**

|  |
| --- |
| 1. We write a class to create objects |

**What is the job of readLine() method?**

|  |
| --- |
| * 1. It waits in the terminal (console monitor) to take the input from the keyboard   2. Whenever you press the enter after typing something then readLine() method takes it and returns as a string |



**How to call instance method?**

|  |
| --- |
| * + 1. We can call instance method by using object only     2. We can all instance method by using reference variable which must contain object address. |

**Why NullPointerException will raise during program execution?**

|  |
| --- |
| If you call instance method by using reference variable which contains null, then we will get this exception. |

**What is Else…if.. ladder and switch statement?**

|  |
| --- |
| If you want to execution one option among many then we can use either else…if ladder or switch statement. |

**What is static import?**

|  |
| --- |
| It is used to import static variables and static methods of a class into our program. If we import them then we can use them directly without using class name. |

**What is the scope and life of local variable?**

|  |
| --- |
| * + - 1. Life means: time between creation and destruction of a variable       2. Local variables are created whenever the declaration statement is executed by JVM and destroyed from stack frame whenever the block where declared execution is completed.       3. Scope: in which block or blocks it is accessible is called as the scope of a variable.       4. Local variables are accessible only within the block where these variables are declared. |

**What is Switch expression?**

|  |
| --- |
| A switch statement can return a value. These kind of switch statements are called as switch expressions  Note: every case must yield a value or thow an exception and it must contain default case |

**What is Looping Statements (or) Iterative Statements?**

|  |
| --- |
| If we want to execute one or more statements repeatedly until the given condition becomes false.  Iterative statements are   * while * do...while… * for * foreach |

**What is the syntax While loop?**

|  |
| --- |
| Syntax-1: while(condition)  stmt-1  Syntax-2: while(condition){  stmt-1;  stmt-2;  …………  stmt-n;  } |

**What is Do…while?**

|  |
| --- |
| In do…while… we will enter into the loop first, after that condition checking will be done.  Syntax-1: do{  Stmt-1;  …………;  Stmt-n;  }while(condition); |

**What is Nested loops?**

|  |
| --- |
| We can write a loop within another loop, it is called as nested loops  Syntax:  for(initialization;condition;update){  stmt-1;  stmt-2;  for(initialization;condition;update){  stmt-1;  ………..  stmt-n;  }  Stmt-3;  ………..  Stmt-n;  } |

What is an array?

|  |
| --- |
| * 1. Till now to store values we declare variables. And memory allocation for the variables will be done in different addresses. So memory allocation is not contiguous. To allocate memory contiguously we use arrays.      * 1. Array is a collection similar data element (data items) stored at contiguous memory locations and shared by a common name.   2. After array creation the default values will be stored in each block.   Advantages of array   * + 1. Accessing/modifying elements is fast     2. We can get/access or modify the elements randomly   **Dis-advantages**   * + - 1. Array size is fixed |

**How to create an array?**

|  |
| --- |
| We have different ways to create an array in Java? |

**In which memory arrays will be created?**

|  |
| --- |
| Arrays will be created in heap memory |

**What will be considered as array address?**

|  |
| --- |
| First block address will be considered as array address |

**What is an index?**

|  |
| --- |
| * + - 1. It is a number given to each and every block of an array. And it always starts with zero       2. By using index, we can access elements faster |

**What are types of arrays?**

|  |
| --- |
| There are 2 types of arrays   * 1D array: it is an array, which is created with single subscript and it contains list of elements of the same type stored in a single row. * 2D array: collection of 1D arrays   (or)  A two-dimensional array is an array of arrays, which is commonly used to represent matrices or tables. |

**Which class object will be created if we create an array in Java?**

|  |
| --- |
| The Java compiler creates an instance of a class that extends java.lang.Object and implements java.lang.Cloneable and java.io.Serializable.  **The class created:**   * Java does **not** explicitly create a named class for the array, but internally, it is represented as a **special class** that extends Object. * The **runtime type** of the array is int[], which is a subclass of Object. |

**What is length of an array?**

|  |
| --- |
| * It is an instance variable existed in every array. Which is used to find the length of an array. |

**How to access static variable or static method?**

|  |
| --- |
| * 1. We can access static variable (or) static method by using either class name or a ref-variable |

**How to access instance variable or instance method?**

|  |
| --- |
| * We can access instance variable or instance method by using object only. |

**When we will get NullPointerException?**

|  |
| --- |
| * 1. We will get NullPointerException when we call instance method or instance variable by using reference variable which is pointing to null. |

**What Is the use of Scanner?**

|  |
| --- |
| * It is a pre-defined class existed in java.util package * This class has many instance methods which are used to get the data from either keyboard (standard input device) or a file. * We can Storing and getting elements from an array by using Scanner class method nextInt() |

**How can you Initializing array during creation itself?**

|  |
| --- |
| Syntax-1: int [] arr={10,20,30,40,50};  Syntax-2: int[] arr=new int[]{10,20,30,40,50};  At the time of reference variable declaration, the subscript can be placed either side of the variable.  Ex-1: int[] arr;  Ex-2: int [] arr;  Ex-3: int []arr;  Ex-4: int arr[]; |

**What is the use of Foreach?**

|  |
| --- |
| For each is used to get the elements from a collection object or an array. |

**What is an anonymous array?**

|  |
| --- |
| It is an array which is created without name and without specifying size. |

**What is Two – dimensional array?**

|  |  |
| --- | --- |
| * 1. A two-dimensional array is an array which contains collection of single dimensional arrays.   It is an array of arrays |  |

**What is the Syntax to create a two-dimensional array**

|  |
| --- |
| Syntax-1: data\_type[][] arr=new data\_type[row\_size][col\_size];  Syntax-2: data\_type arr2[][]=new data\_type[1D arrays count][size of each 1D array]; |

**What is Jagged or Ragged Array?**

|  |
| --- |
| A **jagged array** is an array of arrays where the sub-arrays can have different lengths. Unlike a two-dimensional array (which has fixed row and column sizes), a jagged array allows each row to have a different number of elements. |

**What are Key Characteristics of Jagged Arrays?**

|  |
| --- |
| * It is an array of arrays. * Each inner array can have a different size. * It is useful for scenarios where a rectangular structure is not needed. * In languages like Java and C#, jagged arrays are implemented as an array of references to other arrays. |

**What is matrix multiplication?**

|  |
| --- |
| Matrix multiplication is a method of finding the product of two matrices to get the result as one matrix. It is a type of binary operation. |

**How to multiply two given matrices?**

|  |
| --- |
| To multiply one matrix with another, we need to check first, if the number of columns of the first matrix is equal to the number of rows of the second matrix. Now multiply each element of the column of the first matrix with each element of rows of the second matrix and add them all. We need to do the dot product of columns and rows here. |

**What is the result of multiplication of (2**×**3) matrix and (3**×**3) matrix?**

|  |
| --- |
| The result of multiplication of (2×3) matrix and (3×3) matrix will be 2×3 matrix only. |
|  |

**How to multiply 3**×**3 matrix?**

|  |
| --- |
| Multiply each row of the first matrix with each column of the second matrix and add all to get the first element. Similarly, multiply and add the elements of the two matrices, column and row-wise, to get the elements of the product of two 3×3 matrices. |

**How do we find the multiplication of two matrices?**

|  |
| --- |
| If A is a m×n matrix and B is a p×q matrix, then the multiplication of A and B is denoted by dot matrix, such as: C = AB Thus, C will be an m×q matrix. |

**What is Command line arguments?**

|  |
| --- |
| Command line arguments are arguments, which are given to the main method at the time of giving command to run the program like below.  java First 10 20 madhu 10.40   * 1. in the above command 10,20, madhu and 10.40 are command line arguments   2. these arguments are taken by JVM and place them in a string array and passes that array to the main method as an argument. |

**What the JVM should pass to the main method as an argument?**

|  |
| --- |
| Single dimensional string type array address |

**Note:** A method can have variable length parameters with other parameters too, but one should ensure that there exists only one varargs parameter that should be written last in the parameter list of the method declaration. For example:

**What is the use of @Override annotation?**

|  |
| --- |
| It is used to inform the compiler to verify whether we are overriding a method correctly or not. |

**Why we override a method?**

|  |
| --- |
| If we want base class instance method in child class with different implementation then we override a method. |

**What is method signature?**

|  |
| --- |
| Method signature is a combination of method name and parameter list(parameter type, order and count) |

**What are the rules of method overriding?**

|  |
| --- |
| * The access modifiers must be same or the higher scope. * It should contain same return type and same method signature (parameter list and method name). * We cannot override a static method instead we hide them. * We cannot override a final methods and constructors * We cannot override a private method because it is not no inherited into child class as it is accessible with in the class. |

**What are the access modifiers?**

|  |
| --- |
| * Public: - Accessed anywhere * Private: - within the class * Protected: - with in the package and sub-class of another package * Default (package private ) :- within the package. |

**After overriding the base class instance method in child class, can I call the base class method by using child class object?**

|  |
| --- |
| No, we can’t instead we class call a super class in child class. |

**What is Method overriding?**

|  |
| --- |
| If we want base class instance method in child class again with different implementation then it is called overriding. |

**What is the use of super?**

|  |
| --- |
| * It is a keyword which is used to call the super class constructor in a child constructor. * We can call overridden method in child class by using super keyword. * This represents present object and super represent super class object. |

**What is Simple or single level inheritance?**

|  |
| --- |
| If a class inherits a single base class at a time, then it is called as single level inheritance. |

**What are Forms of inheritance?**

|  |
| --- |
| * Simple or single level inheritance * Multiple inheritance |

**What are Other forms of inheritance (extended forms of simple or multiple inheritance)?**

|  |
| --- |
| * 1. Multilevel inheritance (extension to simple inheritance)   2. Hierarchical inheritance (extension to simple inheritance)   3. Hybrid inheritance (combination of one or more forms) |

**How to inherit a Base class into Child class?**

|  |
| --- |
| By using extends keyword we inherit the base class to the child class |

**What is child class?**

|  |
| --- |
| It is a class which inherits the base class  Child class is also called as derived or sub class. |

**What is base class?**

|  |
| --- |
| If a class is being inherited by another(child) class then it is called as base or parent or super class. |

**What is object state?**

|  |
| --- |
| The data existed in an object is called as an object state |

**What is inheritance?**

|  |
| --- |
| Process of including members (fields and methods) of one class into another class, is called as inheritance. |

**Wht is Polymorphism?**

|  |
| --- |
| * 1. Here poly means many and morphs means forms   2. If we write a method which behaves differently in different situations then it is called as polymorphism. |

**What is method overloading?**

|  |
| --- |
| We can write a method with same name with different signature in a class. It is called as method overloading. |

**What are short cut operators?**

|  |
| --- |
| * += * -= * \*= * /= * %= * ++ |

**Can I assign a value type variable to reference variables and vice versa?**

|  |
| --- |
| No, we can’t |

**What is Explicit type casting?**

|  |  |
| --- | --- |
| **Narrowing Primitive Conversion**  A narrowing primitive conversion may lose information. We have 22 specific conversions on primitive types those are called as narrowing primitive conversions.   * *short to byte or char* * *char to byte or short* * *int to byte, short, or char* * *long to byte, short, char, or int* * *float to byte, short, char, int, or long* * *double to byte, short, char, int, long, or float*   **Note:** while compilation compiler displays an error, if we don’t use the typecast operator. While executing a narrowing primitive conversion runtime error will not occur.  **Note:** there may be a chance of loss of data. |  |

**What is Implicit type casting?**

|  |
| --- |
|  |

**What is the range of character type variable?**

|  |
| --- |
| 0 to 65535  Or  0 to 216-1 |

**If we assign integer literal to char type variable does it support implicit or explicit type casting?**

|  |
| --- |
| If integer literal is within the character range then it support implicit type casting otherwise explicit. |

**What is Widening Primitive Conversion?**

|  |
| --- |
| This conversion is also called as implicit or automatic conversion, because developer no need to use typecast operator explicitly. |

**How many conversions ae there in Widening Primitive Conversions are there?**

|  |
| --- |
| A widening primitive conversion does not lose information. We have 19 specific conversions on primitive types those are called as widening primitive conversions.   * byte to short, int, long, float, or double * short to int, long, float, or double * char to int, long, float, or double * int to long, float, or double * long to float or double * float to double |

**Can I assign a value type variable to reference variables and vice versa?**

|  |
| --- |
| No,we can’t |

**What is method overloading?**

|  |
| --- |
| We can write multiple method with same name with different signature in a class. It is called as method overloading. |

**What is Polymorphism?**

|  |
| --- |
| * 1. Here poly means many and morphs means forms   2. If we write a method which behaves differently in different situations then it is called as polymorphism. |

**What is inheritance?**

|  |
| --- |
| Process of including members (fields and methods) of one class into another class, is called as inheritance. |

**What is object state?**

|  |
| --- |
| The data existed in an object is called as an object state |

**What is base class?**

|  |
| --- |
| If a class is being inherited by another(child) class then it is called as base or parent or super class. |

**What is child class?**

|  |
| --- |
| It is a class which inherits the base class  Child class is also called as derived or sub class. |

**How to inherit a Base class into Child class?**

|  |
| --- |
| By using extends keyword we can inherit a base class into child class. |

**What are the Forms of inheritance?**

|  |
| --- |
| There are two types of inheritance   * Simple (single-level) inheritance * Multiple inheritance |

**What are other forms of inheritance (extended forms of simple or multiple inheritance)?**

|  |
| --- |
| * 1. Multilevel inheritance (extension to simple inheritance)   2. Hierarchical inheritance (extension to simple inheritance)   3. Hybrid inheritance (combination of one or more forms) |

**What is simple or single level inheritance?**

|  |
| --- |
| If a class inherits a single base class at a time, then it is called as single level inheritance. |

**What is the first statement in Constructor?**

|  |
| --- |
| Every constructor will have Super() class constructor calling aas there first statement. |

**If base class has only parameterized constructor what should we do?**

|  |
| --- |
| If base class has only parameterized constructor, then we have to call it explicitly in child constructor. |

**What is the use of super?**

|  |
| --- |
| * It is a keyword which is used to call the super class constructor in a child constructor. * We can call overridden method in child class by using super keyword. * This represents present object and super represent super class object. |

**What is method overriding?**

|  |
| --- |
| If we write base class instance method in child class again, then it is called as method overriding. |

**After overriding the base class instance method in child class, can I call the base class method by using child class object?**

|  |
| --- |
| No, we can’t |

**What are Access Modifiers according to OCJP?**

|  |
| --- |
| According to **OCJP (Oracle Certified Java Programmer)** standards, there are technically **only 3 access modifiers** in Java:   1. public 2. protected 3. private |

**What are levels of access control?**

|  |  |  |
| --- | --- | --- |
| **Access Level** | **Modifier** | **Description** |
| Public | Public | Accessible in any package. |
| Protected | protected | Accessible within the **same package** and **subclasses (even in different packages)** through inheritance. |
| Default (Package-Private) | *(No Modifier)* | Accessible **only within the same package**. |
| Private | Private | Accessible **only within the same class**. |

**What are rules to Follow During Method Overriding?**

|  |  |
| --- | --- |
| **Rule** | **Description** |
| 1. Method Name | The **method name** must be **exactly the same** as in the parent class. |
| 2. Method Signature | The **parameter list** must be **the same** (same type, order, and number of parameters). |
| 3. Return Type | The **return type** must be **the same** or **Covariant Type** (Child class return type is allowed from Java 5 onwards). |
| 4. Access Modifier | - The access modifier **cannot be more restrictive**. ✅ public → public ✅ protected → protected or public ❌ public → protected |
| 5. Static Methods | **Static methods cannot be overridden** (It is called **method hiding** instead). |
| 6. Final Methods | **Final methods cannot be overridden** because they are fixed by the parent class. |
| 7. Private Methods | **Private methods cannot be overridden** because they are not inherited. |
| 8. Exception Handling | The overriding method **cannot throw broader or new checked exceptions** but can throw narrower exceptions. |
| 9. Constructors | **Constructors cannot be overridden**. |
| 10. Synchronization | Synchronization modifier is **ignored** during overriding. |

**What is method signature?**

|  |
| --- |
| Method signature is a combination of method name and parameter list(parameter type, order and count) |

**Why we override a method?**

|  |
| --- |
| If we want base class instance method in child class, with different implementation then we override a method. |

**What is the use of @Override annotation?**

|  |
| --- |
| It is used to inform the compiler to verify whether we are overriding a method correctly or not. |

**How the memory allocation will be done for String literals?**

|  |
| --- |
| String name1=”Madhu”;  For the above statement a new object will be created in string constant pool, if string object with same content is not present already and returns its address.  String name2=”Madhu”;  If the above statement executes old object address will be assigned to name2. i.e.. name1 and name2 contains same address.  String name3=new String(“Madhu”);  Here JVM creates new object in heap memory (not in string constant pool). Without checking about old object with same content. I.e.. name1 and name2 contains same address but name3 contains different address. |

**What is Multilevel Inheritance?**

|  |
| --- |
| * 1. If a child class is inherited by another child class it is called as multilevel inheritance   2. It is an extension to single level |

**What is the advantage of inheritance?**

|  |
| --- |
| Re-usability |

**What is hierarchical inheritance**

|  |
| --- |
| If a base class is inherited by more than one sub class then it is called as hierarchical inheritance.  Inheritance in C++ - TechVidvan |

**What is Has-A-Relationship?**

|  |
| --- |
| If an object contains another object it is called as has a relationship  Example: Engine is an object and Car is an object, here the relationship is Car has engine. |

**What is Weak Has A relationship?**

|  |
| --- |
| **Weak Relationship (Weak Reference)**: A reference type (WeakReference<T>) where the object can be garbage collected if no strong references exist, helping in memory management. |

**What is Strong Has A Relationship**

|  |
| --- |
| **Strong Relationship (Strong Reference)**: A standard object reference where the object is not eligible for garbage collection as long as a strong reference exists. |

**When the objects will be destroyed from heap memory?**

|  |
| --- |
| Whenever the reference count becomes zero then those objects will be destroyed from heap by GC (Garbage Collector). |

**Can we invoke Garbage Collector explicitly?**

|  |
| --- |
| Yes, we can do it by calling gc() method of System class.  Note: gc() is a static method existed in System class. |

**What are accessor methods and mutator methods?**

|  |
| --- |
| Instance methods are divided into 2 categories   * + 1. Accessor methods     2. Mutator methods |

**What is Accessor Method (Getter)**

|  |
| --- |
| A method that retrieves or returns the value of a field without modifying it. |

**What is Mutator Method (Setter)?**

|  |
| --- |
| A method that modifies or updates the value of a field. |

**What is static binding?**

|  |
| --- |
| * At the time of compilation, if compiler decides what should be accessed at runtime it is called as compile time binding or static binding. * It is also called as compile time binding or early binding   (married to sister’s daughter) |

**What is run time binding?**

|  |
| --- |
| * 1. At the time program execution JVM decides what should be accessed or executed. It is called as runtime binding.   2. Runtime binding is also called as dynamic or late binding   (love marriage) |

**What is upcasting?**

|  |
| --- |
| * 1. Process of assigning child class object to Base type reference variable is called as upcasting   2. It is also called as widening reference conversion |

**What is down casting (narrowing reference conversion)?**

|  |
| --- |
| Process of assigning Base type reference variable to Child type reference variable is called as down casting. |

**What Is dynamic polymorphism?**

|  |
| --- |
| At the time implementing polymorphism, if dynamic binding is performed then it is called as dynamic polymorphism. |

**What Is static polymorphism?**

|  |
| --- |
| At the time implementing polymorphism, if static binding is performed then it is called as static polymorphism. |

**What is a jar (Java Application Archive) file?**

|  |
| --- |
| A **JAR (Java Applicaion Archive) file** is a packaged file format used in Java to bundle multiple files into a single archive. It is essentially a **compressed file (ZIP format)** that can contain Java class files, metadata, configuration files, and resources like images, sound files, or libraries. The .jar extension represents these files. |

**How to create a jar file?**

|  |
| --- |
| By using the tool called jar we can create jar files  Example: |

**Can I use the classes and packages existed in jar file in our system without extracting it?**

|  |
| --- |
| Yes, we can but you have to set the classpath for that jar file. |

**How to set the class path?**

|  |
| --- |
| * + 1. Open the edit the system environment variables from windows search box     2. Click on Environment Variables button     3. Click on new button      * 1. Give classpath as variable name and path of the jar file along with it’s name and put the semicolon at the end of the path (see the above image)   2. Ok -> ok -> OK ->   3. DONE |

**Can I use the classes and packages existed in programs folder anywhere within the computer without creating .jar file?**

|  |
| --- |
| Yes we can use then if you set the classpath for the programs folder. |

**What is a classpath?**

|  |
| --- |
| * It is an environment variable which is used to set the path for .class files and packages, so that we can access them from anywhere within the system. |

**What is temporary class path?**

|  |
| --- |
| If you set the classpath within a command prompt, the classes and packages can be used only through that command prompt. If you close the command prompt then class will be gone.    Here %classpath% represents old classpath |

**How can you identify and handle deprecated methods in Java using the -Xlint option?**

|  |
| --- |
| In Java, you can identify deprecated methods using the @Deprecated annotation and the -Xlint:deprecation compiler option. The -Xlint:deprecation flag helps detect and warn about deprecated methods used in the code during compilation.  **javac -Xlint:deprecation Test.java** |