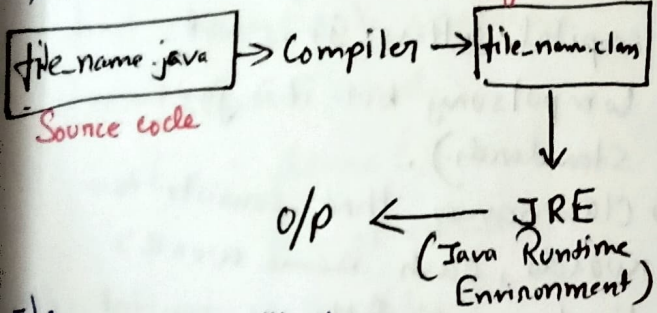
JAVA

1) What is java?
→ Java is a high level programming language.
→ Java is a object-oriented programming Language.

2) How to compile a java program?
→ By using "javac file_name.java".

3) After compilation which file is created?
→ file with ".class" extention is created.

4) Java Architecture.



5) In one java file how many classes can be created?
→ We can create multiple class in one java file.

6) Byte code is created for which class in java?
→ Byte code is ~~created~~ generated for each and every class which is created in java file.

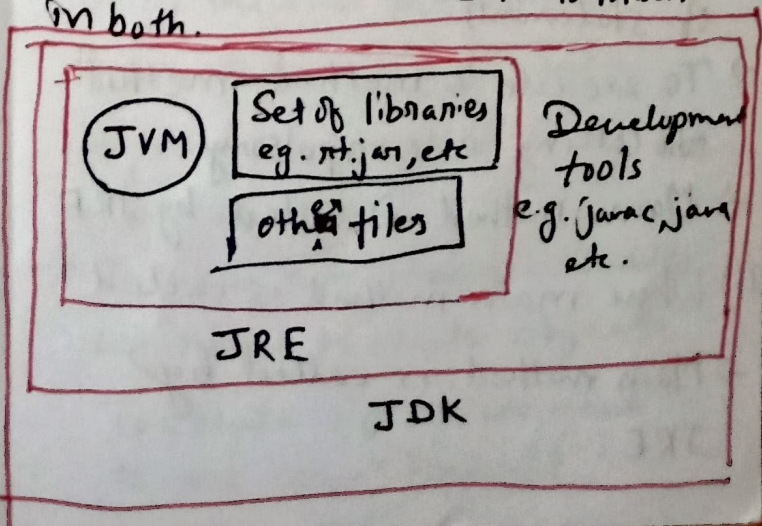
7) How java is platform independent.?
→ Java compiled programs can be executed in any operating system. [Windows, mac OS, linux,... etc].
→ Byte code is platform independent (because of byte code only java become platform independent).
→ Java compiler are not patform independant.

8) Java features
→ Simple easy and familiar
→ Patform independent.
→ Object-oriented
→ Robust (inbilt capability to handle errors)
→ Secure
→ Portable
→ high performance
→ Dynamic
→ Multithreading.

9) What is JDK?
→ Java development kit
→ It provides tools for software development
→ JDK = JVM + JRE
→ It provides ~~development~~ environment for develop & execute java program.

10) What is JRE?
→ Java Runtime Environment.
→ It is a installation package that provids environment to only run (not develop) the java program onto your machine.

11) JVM what?
→ Java virtual machine.
→ JVM is responsible for executing java program line by line.
→ It is important part for both JDK and JRE because it is inbuit in both.



12) What is javac?

→ Javac is primary java compiler included in the java development Kit (JDK). It converts java source code into byte code.

13) Operation performed by JVM?

→ • Loading of code
• Verification of code
• Executing of code
• Providing runtime environment to the users.

14) How many public class is allowed in java file?

→ Maximum one public class is allowed in one java file.

→ Because the file name should be same as that of public class name.

15) What is methods?

→ It is a set of statements which is used to perform a particular task

→ Methods are always created inside the class

→ A method can contain 'n' number of statements.

→ To execute a method, method calling is compulsory.

→ Main method is called by JRE

16) Where main method is called?

→ Main method is called by JRE.

17) What is Identifiers?

→ Identifiers are the names that are given by programmers to java components.

18) Rules to identifiers

→ Not to start with number.

→ No special characters are allowed except ($ and _).

→ Keywords are not allowed to use as identifiers.

19) Rules to class name.

→ class names always starts with capital letters (It is not compulsory but it is java standards).

→ Class names that consists two words, each ~~name~~ word's first ~~name~~ letter is capital letter.

→ Don't give space in bet' two words.

20) Rules to Method name.

→ Method names always starts with lowercase letters. (It is not compulsory but it is java standards).

→ If it consist two words then second words first letter is uppercase.

→ Naming conviction are not mendotory but highly recomended.

21) Java standards:

→ class name should starts with uppercase and remaining characters should be in lower case.

→ If multiple words we should follow camale_case.

Ex: Demo
PrimeNumber.

→ Method name should always starts with lowercase.

→ If multiple words every words' first character is uppercase from second word.

Ex: demo()
demoNumber()

22) What are variables?

→ Variables are the containers which stores single value.

Syntax

datatype VariableName = Value

→ Variables are the temporary memory location.

23) Types of variables.

→ There are 3 types of variables
- Local Variable
- Static Variable
- Non-static variables

24) Local Variable?

→ Variables which are declared inside the method are called local variable

→ Local variables must be intialized before using.

e.g int a = 10;
(int → datatype, a → VariableName, 10 → Value)

25) Static Variables?

→ Variables which are declared inside the class but outside the method with static keyword are called static variables.

→ static variables can be accessed by the object referance but the recomended method is to accessed by the class name.

→ Initialization is not nessesary.

→ For static variables memory will be allocated only once.

→ static variables are alsocalled as class variables.

e.g static int a;

26) Non-static variables?

→ Variables which are declared inside the class but outside the method without static keyword are called non-static variables.

→ To access non-static variables we need to create object. To create object we need to use "new" keyword.

→ Initialization not nessesary.

→ For non-static variable memory will be allocated each and every time of object creation.

→ Non-static variables are also called as instance variables.

e.g
int a;

27) What are operators?

→ Operators are special symbol that perform specific operations on one, two or three operands and then return a result.

28) Types of operators.

→ There are 3 types of operators.

- Unary operators
- Binary operators
- Tarnary operators

29) what are unary operators?

→ Unary operators are those operators that has only one operand. eg ++a, a++, --a, a--.

30) Binary operators?

→ Operators that contain two operand.

eg
a+b, c-d, a/b, etc.

31) Types of unary operators:

→ *Increment

→ Post Increment
(Varname++)

→ Pre Increment
(++ Varname)

→ *Decrement

→ Post Decrement
(Varname--)

→ Pre Decrement
(--Varname)

32) Tarnary operator

→ Operators that contains three operand.

Syntax

```
Variable = cond.? stmt1 ; stmt2;
```

33) Types of operator?

1. Arithmetic operators
2. Assignment operators
3. Relational Operators
4. Logical Operators
5. Bitwise Operators

1. Arithmetic operators
(+, -, *, /, %)

2. Assignment operators
(=, +=, -=, *=, /=, %=)

3. Relational Operators
(==, !=, >, <, >=, <=)

4. Logical Operators
(&&, ||, !)

5. Bitwise Operators
(~ → Bitwise complement,
<< → left shift
>> → Right shift
>>> → Unsigned Right shift
& → Bitwise AND
^ → Bitwise XOR
| → Bitwise OR.

34) Types of loops in Java.
- for
- while
- Do-while
- for each

35) What is difference between for and while loops.
→ If the number of iterations are known we use for loop.
If the number of iterations are not known then we use while loops.

36) What is do-while?
→ It executes atleast once.

37) Difference if-else and switch?
If-else checks equality and logical expressions.
whereas switch checks only equality.

38) Range of byte, short, int, long.
byte → -128 to 127
short → -32,768 to 32,767
int → -2^{31} to $2^{31}-1$
long → -2^{63} to $2^{63}-1$

39) What is type casting?
→ The process of converting one data type to other data type is called as type casting.

40) What is widening
→ Converting smaller primitive type to bigger primitive type.
→ Widening is implecit.

41) What is narrowing?
→ Converting bigger primitive type to smaller primitive type is known as narrowing.
→ narrowing is explecit.

Note*
Java supports auto widening.

42) What is OOPs?

- OOPs (Object Oriented Programming) tries to map code instructions with real world making the code short and easier to understand.
- Solving a problem by creating objects is one of the most popular approaches in programming. This is called object oriented programming.

43) What is DRY?

DRY stands for: ~~do not~~

Don't Repeat Yourself.

44) Explain public static void main (String args[]) in java?

public static void main(String []args)

public: public is a access modifier, which is used to specify who can access this method. public means that method can be accessable by any class.

static: It is a non-access modifier. It is mainly used for memory management. main() method is made static in java so that it can be accessed without creating the instance of a class.

Void: It is the return type of the method. Void defins the method which will not return any value.

main: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occure.

String [] args: It is the parameter passed to the main method.

45) Why java is not ~~object~~ 100% object oriented.

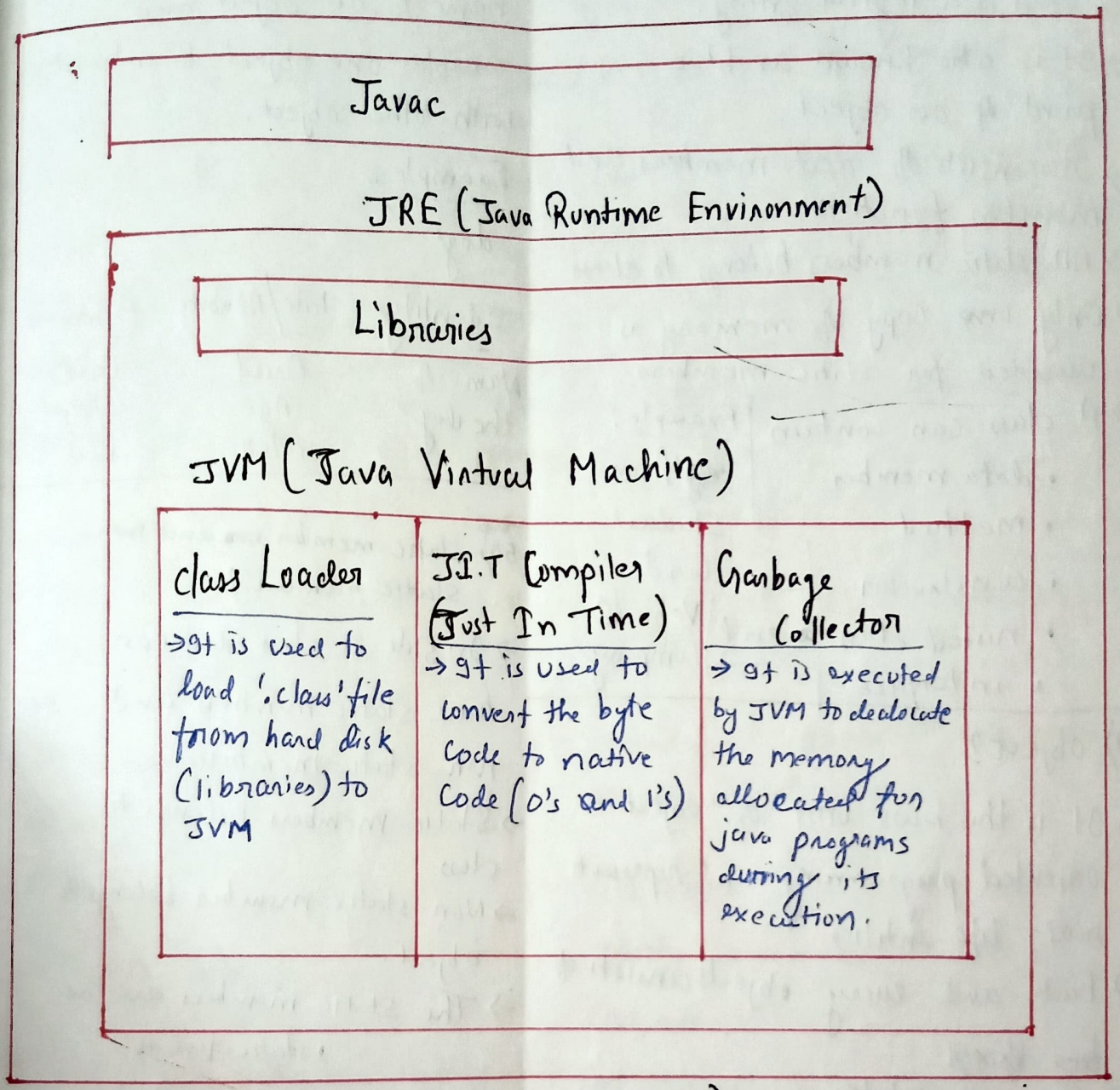
→ Because it makes use of eight ~~peer~~ primitive data types such as byte, short, int, long, double, float, boolean, char.

46) Difference between break and continue statement.

→ Break is used to exit from the loop and continue is used to skip the current iteration.

47)

47) JDK Architecture :



JDK (Java Development Kit)

→ JDK is platform dependent, JVM and JRE also.

→ JRE allocates memory inside RAM for execution of java programs. It consists of libraries and JVM.

→ Libraries are the java programs available in hard disk. It can be pre-defined or user defined java programs.

→ JVM is used to execute the byte code.

48) What is class?

→ class is a user defined data type
→ class is a logical entity.
→ It is also known as blue print of an object.
→ It consists of data members and member functions.
→ All static members belongs to class
→ Only one copy of memory is allocated for static members.
→ A class can contain
- data member
- method
- constructor
- nested class and
- interface

Example:
Animal
Student
Bird
Vehicle
Company

49) object?

→ It is the basic unit of object-oriented programming and represents real-life entities.
→ Each and every objects consists of
- state
- Behaviour
- Identity

• State: It is represented by attributes of an object. It also reflects the property of an object.
• Behaviour: It is represented by methods of an object. It also reflects the response of an object with other objects.
• Identity: It gives a unique name to an object and enables one object to interact with other object.

Example:

dog

Identity	State/Attribute	Behaviours
Name of the dog	Breed	Bark
	Age	Sleep
	Color	Eat

50) Static member and non-static member.?

→ In side a class we can have static members and non-static members.
→ Static members belongs to class
→ Non-static member belongs to object.
→ The static members are the
- static variable
- static method
- static block

→ Non-static members are
- non-static variable
- non-static method
- non-static block

→ Static members's memory is allocated at the time of class loading and a class is loaded only once.

→ ~~Non-static memory is created at the time of object creation~~
→ Non-static members's memory is allocated at the time of object creation (Instantiation) and objects can be created 'n' number of times.
→ Always class is loaded first and then objects are created.
→ Static method can access static data members directly, but cannot access non-static members directly instead we should create an object and access.
→ Non-static methods can access both static & non static data members directly

5) class and object difference?

class	object
→ class is blueprint of an object. It is used to create objects.	→ An object is an instance of the class.
→ No memory is allocated when a class is declared.	→ Memory is allocated as soon as an object is created.
→ A class is a group of similar objects.	→ An object is a real world entity such as a book, car, etc.
→ class is a logical entity	→ An object is a physical entity.
→ A class can only be declared once.	→ Objects can be created many times as per requirement
→ An Example of class can be a car.	→ Objects of the class car can be BMW, Mercedes, Ferrari, etc.

52) Referance variable?

→ A referance variable is used to store address of the object.
→ clases, interface, arrays, enumerations, and annotations are referance type in java. Referance variables holds the objects/values of referane type in java.
→ The default value of referance variable is Null which means there is no object to point.
→ The referane variable can at most point to one object.
→ The object without a referance is known as abonden objects or anonymous objects.
→ Anonymous objects can be reused until and unless it is garbage collected.
→ An object can have more than one referance.
→ Changes made by one referance will affect to all the referances made for the same object.

---

53) This Keyword ?

→ 'this' is a referance variable that refers to the current object.
→ 'this' Keyword can atmost point one object at a time
→ 'this' Keyword can be used implecitly and also explictly
→ It can be used only inside non-static method but cannot be used inside static method.
→ It can also be used inside the constructor
→ If non-static variable and local variable has the same name 'this' keyword is mandatory to be used to point to the non-static data member.

54) Blocks ?

→ In java the { and } is known as blocks.

→ Blocks are also called as initializers.

→ In order to perform any operations while assigining values to instance data members, an initializer block is used.

→ It runs every time whenever the object is created.

→ The intializer block contains the code that is always executed whenever an instance is created.

→ There are two types of block namely

- Static block
- Non-static block

→ Static block's memory is allocated and executed during the class loading. static block is created using 'static' keyword.

→ Since class is loaded only ones static block is also executed only ones.

→ Static block is used to intialize static variables.

→ If more than one static block are defined for the class, they are executed in sequential order.

→ Non-static blocks are also known as non static intializer.

→ Non-static blocks memory is allocated and executed during object creation.

→ Non-static block is executed for every ~~block~~ object created.

→ In Non-static blocks we can access both static and non-static members.

→ non-static block do not have any keyword.

→ If more than one non-static block is defined then they are executed in sequential order.

55) Method Overloading?

→ It is defined as different methods have same method name with different signatures (or) Parameters.

→ Methods can be overloaded based on three types.

- Length of arguments.
- Type of arguments
- Order of occurance of arguments.

→ Method Overloading is all about performing the same task in different ways.

→ Method overloading is also known as compile time polymorphism, static polymorphism (or) Early binding.

→ Method overloading can be achived only with respect to argument, not based on return type.

56) Can we overload static methods?

→ Yes, we can overload static methods.

→ We can have two or more static methods with same name but difference in input parameters.

57) Can we overload method that differs by static keyword.?

→ No we cannot. overload two method in java if they differ only by static keyword.

→ Member function declarations, with the same name and name parameter-type-list cannot be overloaded if any of them is a static member function declaration.

58) Can we overload main method in java?

→ Like other static method, we can overload main method in java.

54) Does java support operator overloading?

→ java doesn't allow user defined overloaded operator, but internally java overloads operator. for example: + is overloaded for concatenation.

---

60) Constructor?

→ Constructor is a special method in java which is used to initialize the object during its creation.

→ Constructor is a set of statements that is executed when object is created.

Syntax:

```
AccessModifier ClassName (){
    ≡
}
```

→ Constructor has same name as the class name.

→ Constructor does not have any return type.

→ Types of constructor
- Parameterized constructor
- Non-parameterized constructor
- default constructor.

→ If a constructor is defined with return type then it is no more a constructor insted it become a non-static method.

→ When a class is created it is not a empty class insted JVM executes a constructor known as default constructor.

```
class A {
    //default constructor
}
```

61) Constructor overloading?

→ Having same constructor name with different arguments

→ Constructor is overloaded based on length of argument, type of arguments, and order of occurance of argument.

62) Copy constructor?

→ Copying the state of one objects to a new created object with in the same class is known as copy constructor.

→ Constructor overloading is required to achieve copy constructor.

```
class A {
    int i;
    public A(int i){
        this.i=i;
    }
    public A (A a ){
        this.i = a.i;
    }
}
```

```
class CopyConstructor {
    p.s.v.m() {
        A a1 = new A (15);
        A a2 = new A (a1);
    }
}
```

a2 → Current Object

a1 → given object

63) Constructor chaining?

Call to this : this()

→ It is used to call a constructor from another constructor with in the same class.

→ It can be used only inside a constructor but not in method

→ It can atmost call only one constructor.

→ It must be the first line inside constructor.

→ It is used to pass the value from one constructor to another constructor with in the same class.

→ Calling a constructor from another constructor with in the same class is known as constructor chaining.

63) Association:

→ A class communicating with another class is known as association

→ Association can be achieved either through object or class names

→ Association achieves reusability.

65) Aggregation?

→ Creating referance of the class in another class is known as aggriegation.

→ The existance of the container object is not depended on the existance of contained object.

66) Composition?

→ composition is a type of aggregation.

→ The existance of the container object is completely dependent on the existance of contained object.

67) final?

→ 'final' is a keyword in java.

→ It is used to provide restrictions decisions can't be changed

→ It is used on a variable, method (Non-static), class.

→ final declared for a variable converts the variable into a constant.

→ final keyword can be used for local, static, non-static variables.

→ A variable declared as final cannot be reassigned.

→ It do not have any default values, they are allowed to initialize only once.

→ final variables can be assigned but cannot be reassigned
→ final methods can be extended but cannot be overridden by sub class.
→ final classes cannot be extend or inherited.

68) Inheritance

~~→ According the p~~

→ Accuring the property of parent class to child class without changing the property of parent class is known as inheritance
→ It is also known as 'Is-A' relationship
→ It is used to achieve reuseability.
→ It avoids object creation.
→ It is unidirectional.
→ The class getting inherited is known as base class, superclass, parent class.
→ The class which inherits is known as derived class, sub class, child class.
→ Extend keyword is used to achieve inheritance
→ It is always recommended to create the object of child class because child class object can access its own properties as well as parent class properties.

Types of inheritance

1) Single level
2) Multi level
3) ~~Multiple~~ Hierarchical
4) Multiple
5) Hybrid.

Single level inheritance

→ Having only one level of dependency
→ Always mandatory to create object for child class.

Multiple level inheritance:

→ Having more than one level of dependencies.
→ Always mandatory to create object for grand child class.

Hierarical inheritance

→ For a parent class having more than one child class.
→ Always mandatory to create object for child class.

Multiple inheritance

→ For a child class having more than one parent class
→ Multiple inheritance, we can't achieve through classes because assume that I have a, b, c classes if I create object for c class I can't access the property from parent class, if you are trying to access JVM will go for confusion whether which class property, need to access This we can't achieve it through classes but we can achieve it through inheritance. This problem is known as ambiguity problem. (or) diamond problem

Hybrid inheritance

→ The combination of hierical inheritance & multiple inheritance
→ It is not supported through classes.

69) Method Overriding?

→ Method overriding, having the same method name with same signature (arguments) is known as method overriding.

→ 'Is-A' Relationship mendatory

→ Overriding can be achieve only for non-static methods, Not for static methods or constructor.

→ If the sub-class is not intrested by the implementation provided by super class using the same method name and arguments. This is known as method overriding.

70) Annotation?

→ Annotation in java is used to provide special instruction to JVM

→ Annotation in java can be used for a class, data member, methods, constructors & interface.

→ Annotation starts with '@' special character.

71) @overriding?

→ @ overriding annotation is available in 'java.lang' package

→ @ override annotation is used to inform JVM to strictly check method overriding.

72) Access modifiers in java?

→ It is used to achieve security or provide visibility for member of the class

→ They are of 4 types

- Private
- protected
- public
- package (default)

* Private Access modifier:

→ The members declared as private can be accessed only within the class where it is created.
→ It is used to achieve security.
→ Private members cannot be accessed by the sub classes and other classes of the same package or other package.
→ To access private members, public methods are defined.

* Public Access modifier:

→ It is used to provide visibility for the members
→ If a member is declared as public, it can be accessed within the entire project.

* Package access modifier (default access modifier)

→ If a member is not declared any access modifiers by default it belongs to package
→ If member declared as package can be accessed only within the package, not in other package.

* protected access modifier

→ If a member is declared as protected, it can be accessed within the package.
→ To access the ~~package outside~~ protected members outside the package "IS-A" Relationship is mandatory.
→ To access protected members outside the package sub class objects should be used.

	Private	Package (default)	Protected	Public
Same class Same package	✓	✓	✓	✓
Sub class same package	✗	✓	✓	✓
other class same package	✗	✓	✓	✓
sub class other package	✗	✗	✓	✓
other class other package	✗	✗	✗	✓

73) Object Class?

→ It is the super most class of the entire java class hierarchy.

→ It is a non-final class. (means it can be extended)

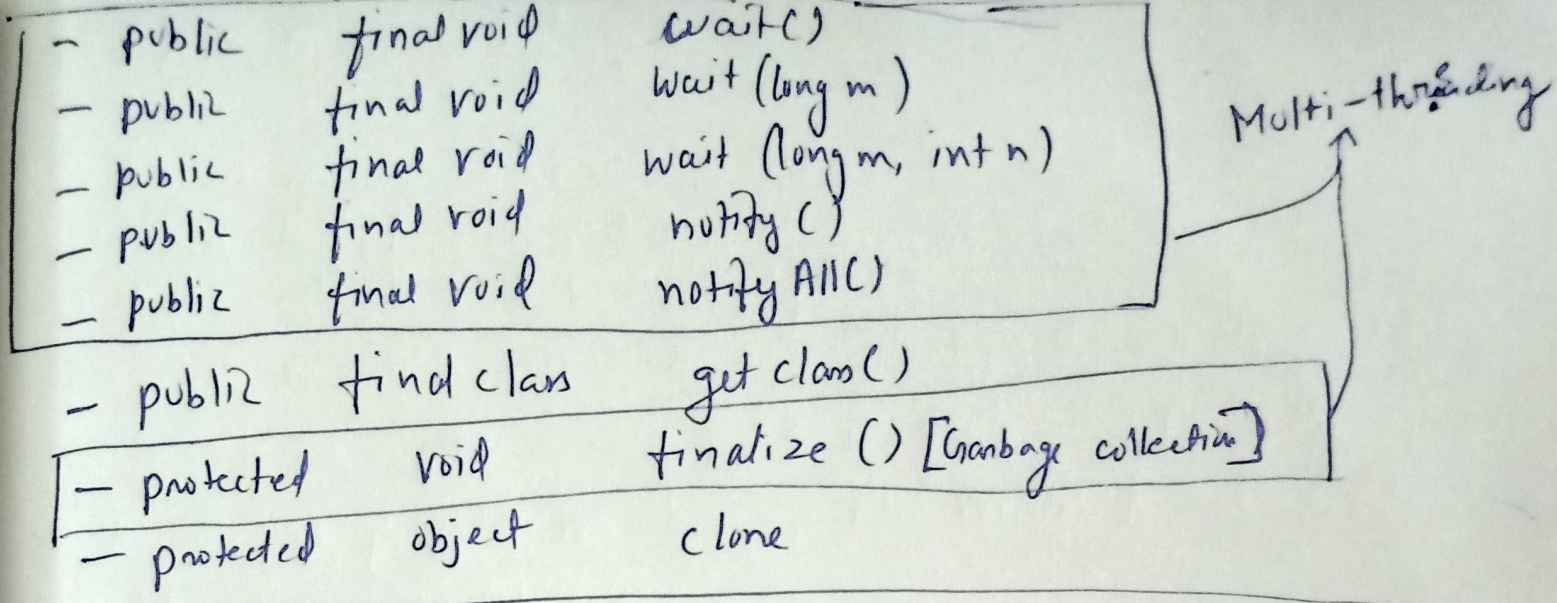
→ It is available in java.lang package

→ Every class developed or created by default extends object class.

→ object class has only one constructor that is no argument constructor. (default constructor).

Methods of object class:

- public String toString()
- public Boolean equal(object obj)
- public int hasecode()
- ~~opu~~



74) Overriding toString()?

→ To string() belongs to object class and when used in our classes it returns the address of the object.

→ toString() should be overridden in our classes to provide the string representation of the state (content) of the object