ARYAN SCHOOL OF ENGINEERING & ECHNOLOGY

BARAKUDA,PANCHAGAON,BHUBANESWAR,KHORDHA-752050

S.C.T.E. & V.T.
ODISHA

LECTURE NOTE

SUBJECT NAME- OBJECT ORIENTED METHODOLOGY

BRANCH-COMPUTER SCIENCE ENGG.

SEMESTER-3RD SEM

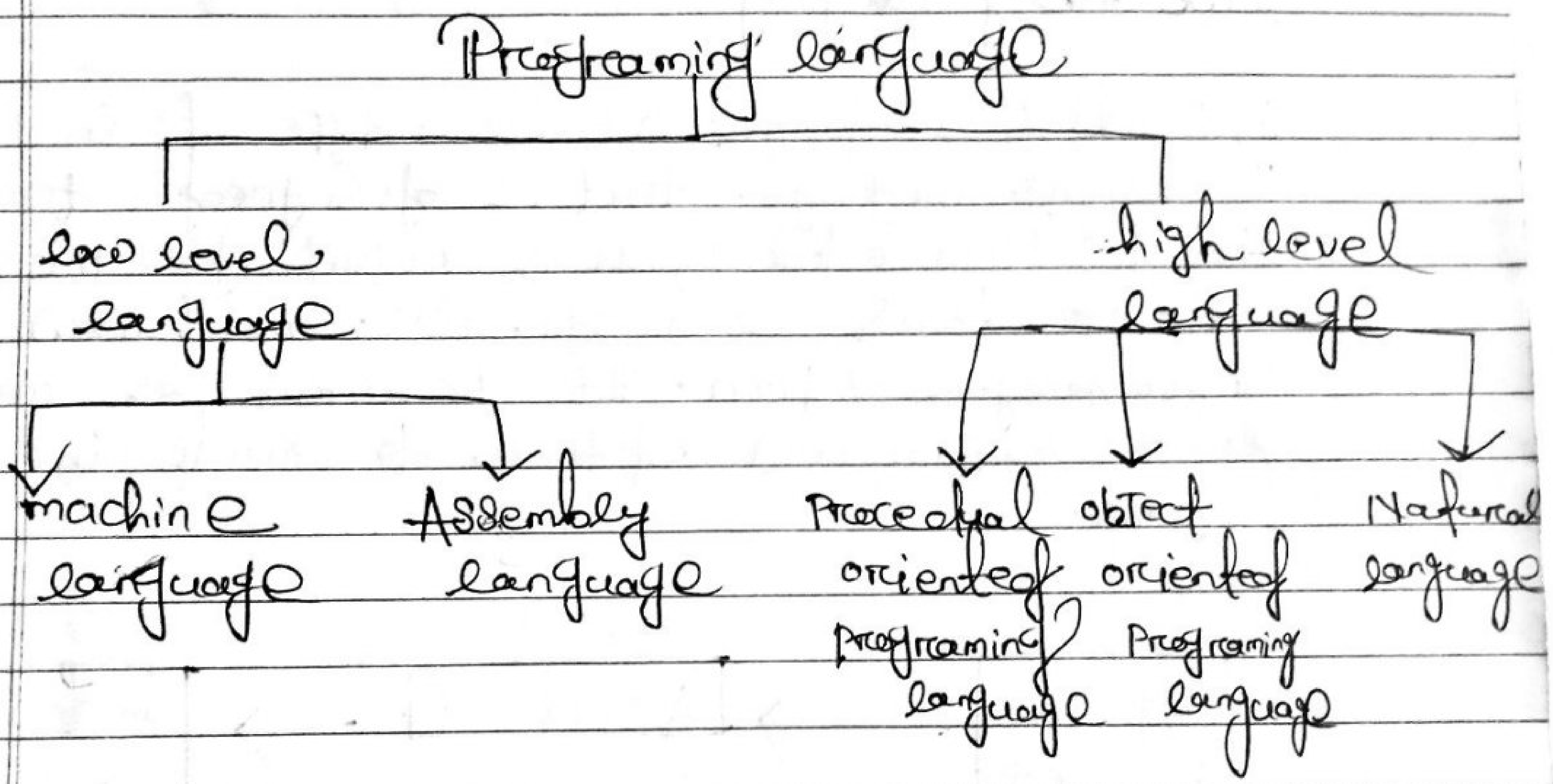
ACADEMIC SESSION-2022-23

PREPARED BY- AMITAV PARIDA

Programing Language :-

* A Programing language is a Computere language, i.e used by Programmers to Communicate with the Computere.

* It is a set of instruction written any in a specific language (C, C++, Java, Python). to Perform a specific task.



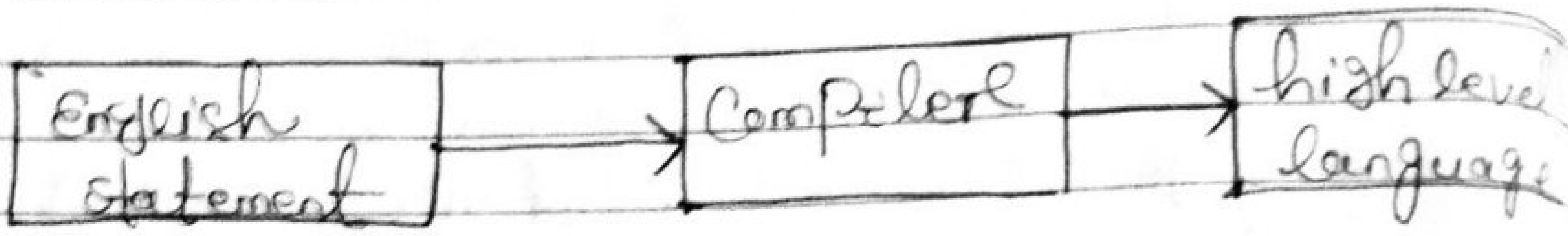
low level language

low level language is a machine depend as (0 and 1) Programing language. The Processere reans low level Programing language directly without any need of Compailere and interpeter.

machine level language

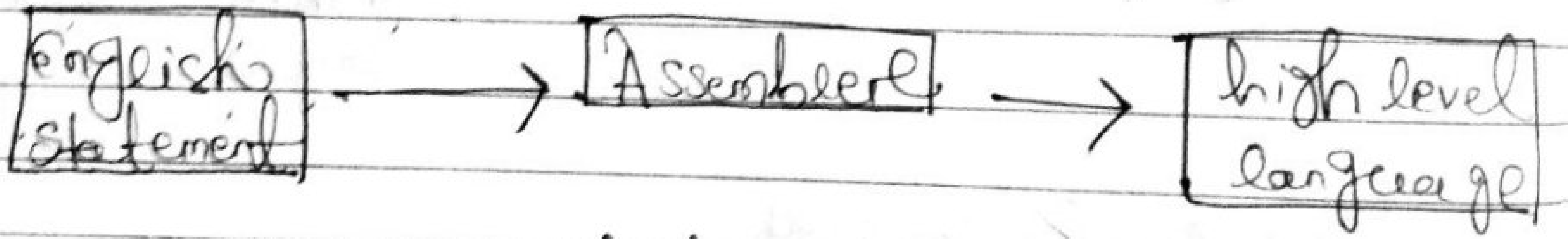
machine level language is a type of low level Programing language and it is also Called as machine ~~Code~~ Code ore object Code. It is helps the Programmere to excute the Programmers fastere

~~tan~~ than the high level Programming language.



Assembly language

Assembly language is also a type of low level Programing language that is designed for specific procedures, It is represents the sets of instructions in a symbolic and human understandable from. It requires less memory less execution of time to excute the Program.



~~Procedual oriented~~

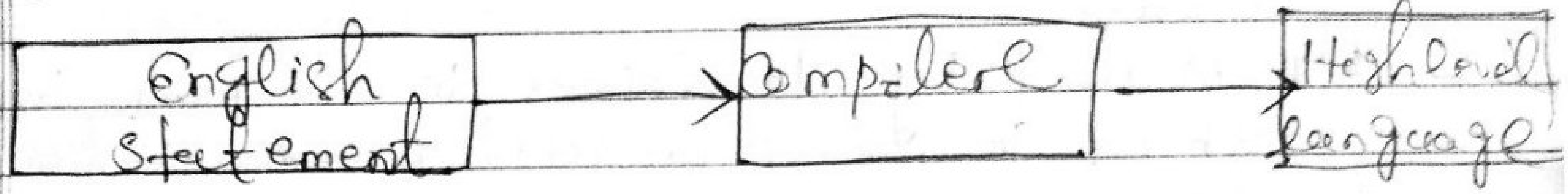
~~Procedual orinted Programing language:~~

~~Procedual orinted Programing language is derived from structer Program based when the procedual all Concept. It is divides a program into small Procedual called~~

High level language :-

* High level language is a desingned fore developing usere friendley software programs and websites. The programing language required a compilere and an interepreter to translate the program into machine language (0 and 1).

ex C++, C, Java, Python.



Procedual oriented Programing language (POP)

* It is derived from structure programing and based upon the procedure Cav concept, It devides a program into small procedures called Rotines and functions. It helps the programmers to easily talk the program ~~f~~ flowse and code can be reused in different parts of the program.

Real world entity

OBJECT ORIENTED PROGRAMMING LANGUAGE (OOP)

object oriented programing language is based upon the object in this programing language. programs are divided into small ~~&~~ parts called object. It is used to implement the real ~~wo~~ world entities like Inheritance, Polymorphism abstraction, encapsulation etc.

→ oop method is commonly used to develop software packages.

8208682

-> The main advantages of oops is fastere, easierd to execute, maintain, modify etc. as usere as devug (c++, Java, Phython).

Natural language

* Natural language a part of human language such as engeish, Russia. It's used by machine understand, manipulate and interpret human's language.
* It is used by developers to perfrom task such as translaction, automatic summarication etc.
* main advantages of Natural language is that it heeps user to ask Question in any subject and directly respond with in second.

Middle level language

* Middle level language ~~uses~~ lies between the low-level programming language and high level language.
* It is programing also known as intermediate programming language and pseudo [illegible] language.
  e.g :- c, ~~Html, Java~~, ~~Python~~, c++

object oriented programing concept:-
object oriented programing that foure basic concepts:-
1. encapsulation
2. Abstraction
3. Inheritance
4. Polymorphism

① Encapsulation :-

-> Binding both code and data members all together into a single unit is known as encapsulation. A Java code is the example of encapsulation and Java bean is the freely encapsulated class because all the data members are private here.

② Data abstraction :-

* It is the process of hinding the back ground details and show only the functionality.
* It is achived by abstract data types.
* It is achived by using Abstract class.
* 0-100% data abstraction is normally achived by data abstraction.

e.g-> suppose stack is an abstract da types and it contains some methods like pu pop peek etc and how the operation is carr going on this is unknown to the end user.

③ Inheritance :-

* It is the process of creating a new type from an existing type.
* when one object acquires all the properties and behaviour of present ob is called as inheritance.

e.g;— plane and fightere plane are two diffrent classes the instance of fightere plane class will be a new instance of prane type it has been inheritance

(4) Poly morphism ÷

-> one task is going to be performed in different ways.

-> Achived two different ways compile time and runtime.

-> solved using function overloading and function overriding.

CLASS ÷

* class is a blue print of an object or logical entity. define

* It is the keyword to creat user defined data type.

* It is a class name present in side Java or pakage.

* All total of it we will say a class is normal an encapsulated area of properties, value and method.

OBJect ÷

* It is an real owned life entity which an existance.

* from language point of view it is a simple variable.

* from programmers point view it is a simple memory location.

* object is the supper class of Java
object present on side Java lang pakage

Redundancy:-

* Inheritance is the good feature for data redundancy if you need a same functionality is multiple class you can write a common class for the same functionality and inherit that class to subclass.

Easy maintenance :-

* It is easy to maintain and modify existing code as new object can be created with small differences to existing ones.

Security:-

using data hiding and abstraction only necessary data will be provided thus maintain the security of data.

Application of oops

Client - server system:

* object - oriented client - server system provide the IT infrastructure, creating object - oriented client server internet application.

Object oriented Data base:

* They are also called object data base management system (ODBMS) these data base store objects instead of data such as

real, numbere, bersenal integers.

Real time system design -

* Real-time system inherent compleoxities that make it difficult build them object oriented techiques make it easire to handle those complexities.

Simulation and modelling system.

* It ~~cof~~ is difficult to model complex system due to the varying specification of variables, object oriented programing provides an alternative approach for simpliteng these complex modeling system.

Hypertext and Hypermodel

* opps also helps in laying out a frame work for hypertext. Hypertext is similar to regular text as it can be stored, searched and edited easily.

CIIM / CAD / CAM

* OPPS can also be used in manufactoring and design applications, as it allows people to reduce the effort involed. OPPS make possible for the designs and engineers to produce these flow charts and blue prints accurately

~~Encapsulation~~- Benefits of OOPS :-

Re-usability:-

write once ~~andus~~ and usere it multiple ~~times~~" you can achives this by using class.

Redundancy:

Inheritance is the chief feature fore data redundancy if you need a same functionality in multiple class you can write a common class for the same functionality and inherit that class to subclass.

Easy maintenance :- It is easy to maintain and modify existing code as new object can be created with small differences to existing ones.

Security:- using data hiding and abstraction only necessary data will be provided JAVA PROGRAMING thus mantain the security of data.

★ write a Program to calculate the addition of two numbers

```
-> class Test
{
Public static void main (Strings args[])
{
 int a=7;
 int b=6;
System.out.Prentln (a+" "+b);
            int sum = a+b;
System.out.println ("The sum of two nos:"+ sum)
}
}
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

Scanned with OKEN Scanner

classmate
Date ______
Page ______

* write a Program to calculate the modulus of two numbers.

```
> class Test
{
Public static void main (string args[])
{
 int a=5;
 int b=4;
System.out.println (a+" "+b);
      int mode=a%b;
System.out println ("modulus of two nos:" +mode);
```

Path test

* Temporary way to set JDK 1.8
→ go to cmd
→ then TYPE
set path = "copy address of 1.8 JDK"
→ Then enter

go to colisk

copy the address of Java 1.8 bin folder

* Delete
Control + a = select all.

* Permanent: way to set JDK 1.8
step-1 - go to the PC
" -2 - Then right click
" -3 - go to properties
" -4 - then go to the enviromental variable
" -5 - Then click new

variable value / Path
Address Address of bin folder 1.8 JDK

Java

Java - Java is a Programming language and an Platform independent.

→ Java is high level, robust and secure Programming language.

→ It is first developed sun microsystem in a year 1995, James Gosling" is known as the father of Java.

→ Before Java is named was "OAK"

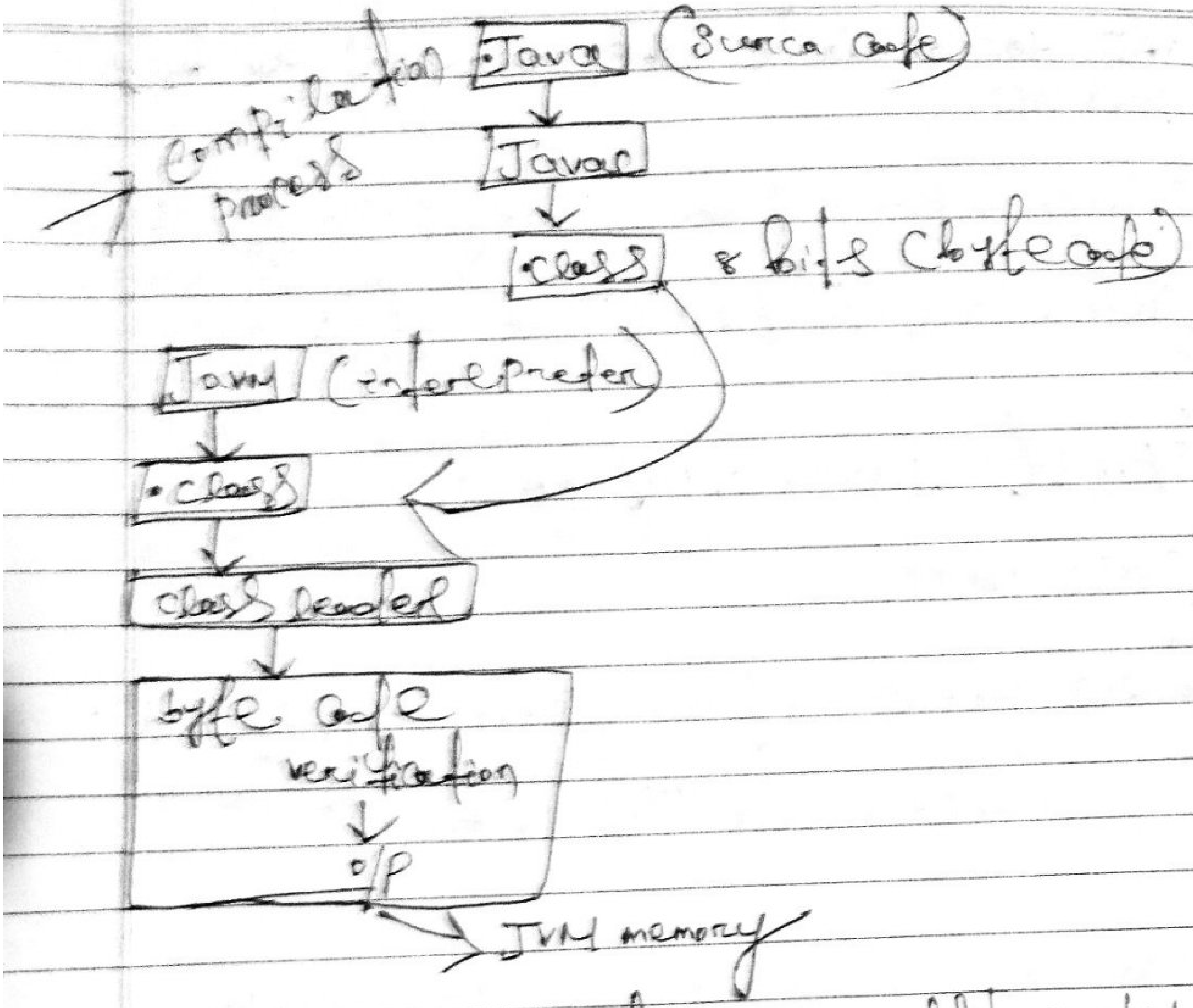
→ Science "OAK" was already a registered company so, James Gosling and his team changed the named from up to Java.

Execution model of Java

→ An execution model specifies how work takes place. every Programming language has a execuation model, which is specified as the part of language specification.

→ In order to understand the behaviour of a Java program, we must have some model, model of how such a program is execute, here we represent by the Java execution model, which explains the step by step excution of a Java program.

Scanned with OKEN Scanner



★ WAP to calculate the addition of two numbers by using parseInt

```
→ class Test
{
public static void main (String [ ]a)
{
   int i,J,sum=0;
 i =Integere.parsent (a[0]);
 J = Integere.parsent (a[1]);
   sum = i+J;
System.out.println ("Addition of two nos;"+sum)
}
}
```

- wap to calculate the size of the byte

```
-> class Test
{
public static void main(String[] args)
{
System.out.println(SIZE.byte);
}
```

First Java Program.

```
class Test
{
public static void main(String args[])
{
System.out.println("Hello Java");
}
}
```

-> save the whole file as Test.java

-> To compile:

    javac Test.java

-> To execute

    java Test

variables and data types :-

variable

-> A variable is a container which holds the value while the Java program is executed. A variable is assigned with a datatype.

-> there are 3 types of variable.

Scanned with OKEN Scanner

1 [illegible]

2 [illegible]

4 [illegible]

4 [illegible]

8 [illegible]

8 [illegible]

Local variables

[illegible]

[illegible]

[illegible]

[illegible]

2. Global variables

[illegible]

[illegible]

[illegible]

string collage "Anyan"
}
using static variable
class student
{
int reall no; //Instance variabll
String name;
Steetil string collage = "Anyan"
}

Instance variable / Non-static

→ It the variable is the part of class directly but declare without any static key word is known as non-static variable

object

→ The variable only can be called by object name.

→ But we can call the static variable by class name, object name and directly. Directly possible with in the same class

The Java virtual machine

→ Java virtual machine (JVM) is an engine that Provides runtime environment to drive the Java code or application to converts Java byte code into machines language JVM is a part of Java Runtime environment (JRE). In othere programing language, the compilere produces machine code or a particular system. However, Java compilere Produces code for a virtual machine known as

Java virtual machine

How JVM works

-) First, Java code is compiled into byte code this byte code gets interpreted an different machine.

-) Between host system and Java Source, Byte Code is an intermediary language.

Final variable

-) If the variable is final then It is a Constant in Java

-) Final variable should be initialize at the time of declaration Cannot be re are modify

final variable

```
class Test
{
final static int a=20;
     final int b=10;
Public static void main (String [] args)
{
  system.out.println(a);
      a=21; //error
```

Datatypes	Defalt value	Default size
1) Boolean	→ false	1 bit
2) chare	→ \u0000	2 byte
3) byte	→ 0	1 byte
4) short	→ 0	2 byte
5) Int	→ 0	4 byte
6) long	→ 0L	8 byte
7) Float	→ 0.0f	4 byte
8) double	→ 0.0d	8 byte.

* Byte:- → The byte data types is an 8 byte sine 2's complement integere.
→ It has a minimum value of -128 and maximum value of +127

* SHORT:- → The short data types is a 16 bit signe 2's complement integere.
→ It has a minimum value of -32768 and maximum value of +32767.

* Integere:-
→ By default the integere data types is 32 bit seign 2's complement integere
→ which has minimum value -2147483648 and maximum value +2147483647

* Long:-
→ The long data type is a 64 bit 2's complement integere
→ It has a minimum value of $-9.223 * 10^{28}$ and maximum $+9.223 * 10^{28}$

* Float:-
→ The float data type is a single precision 32 bit floating.
→ It has a minimum -2147483648 maximum +2147483647

Double ÷

→ The Doubble data type is a priecision 64bit floating point.

→ It has a minimum $-9.23*10^{28}$ and maximum $+9.22*10^{28}$

Chare ÷

→ The chare data types is single 16 bit chare.

→ It has minimum -32768 and maximum +32767

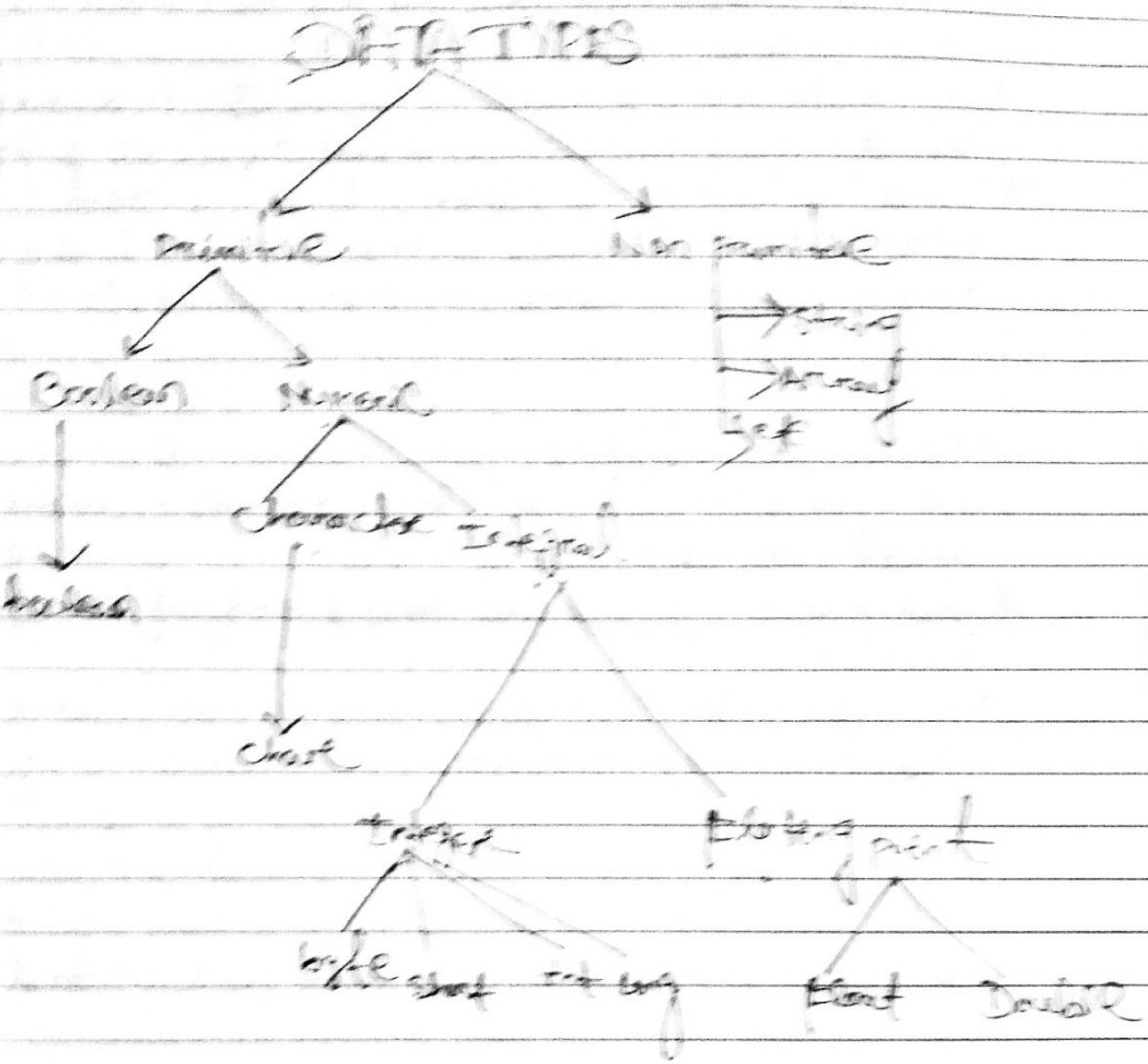
Q Write a program to known the range of data types?

A
```
Class Test
{
public static void main (String [ ] args)
{
system.out.printen (Byte.MaX_VALUE)
                          (underscore)
}
}
```

*

```
class Test
{
public static void main (String args [])
{
  Int n=29;
  if(n%2==0)
  {
   system.out.printen ("Numbere is even")
  } else
  else
}
```

System.out.println



Numeric literals

→ Numeric literal are two types

(i) exact

(ii) approximat

i) Exact:- An exact numeric literals is a numeric value without any decimal Point such as 65, -226, -223. using the Java on ~~exat numeric literals~~ support numbers

[illegible] specify an integral numeric literals
~~[illegible]~~
which [illegible] in the range of Java long

(4) Floating :-

→ We [illegible] numeric literals in a numeric value in scientific notation such as 5.4e5.

→ Using the [illegible] of Java [illegible] literals approximate numeric literals [illegible] the [illegible] in range of Java double.

Character Literals :-

→ For character data types we can specify the literals in three ways.

(i) Single quote :- we can specify the [illegible]

we can specify the literals to char data type as single character within single quote

we can specify the character literals as

→ which represent character value of a can be specified and that integral literals either hexadecimal octal and decimal form, but allowed the range 0 to 65535

we can specify the character literals in unicode representation

1) single quote: we can specified the literals to chare data type as single charactere within single quote.

Syntax:

char ch = 'a';

2/ **chare literal as integral literal:**

we can specified the charactere literals as Integral literals which represents unicode value of the character and that integral literals can be specified eithere in decimal, octal, and hexa-decimal forms. But allow the range is 0 to 65535.

Syntax:

char ch = 062;

3/ unicode representation:

we can specified the charactere literals in unicode representation '\uxxxx'. Here, xxxx represents hexa-decimal numbers.

Syntax:

char ch = '\u0061';

4/ escape sequence: Every escape charactere can be specified as charactere literals.

Syntax:

char ch = '\n';

* WAP to illustrate the application of char:

```
class Test
{
public static void main (String [] args)
```

Scanned with OKEN Scanner

[illegible]

[illegible]

[illegible]

Scanned with OKEN Scanner

ex 2

```
class test {
Public static void main (String [] args)
{
String Temp = "Ram is a good boy";
        String substring = Temp.substring(0,4);
system.out.println (substring);
  }
  }
```

Arrays ÷

Array is an object in Java which contents similar type of data in continuous memory location.

Sintax ÷ → subscript operator

```
or [ data-type[] var.name;
   [ data-type var.name [];
Ex - int a[];
```

```
class Test
{
 Public static void main (String args)
  {
  int a = 10, b=20, c=30, d=40 t=90
    int a[] = {10,20,30,40,50};
  system.out.println(a[3])
  }
  }
```

Fixed size static Array :- (deleration)

```
  ~~Classtest~~ test
 public
class Test
{
Public static void main (string[] args)
{
  int a[] = new int [5];
   a [0] = 10;
    a [1] = 20;
    a[2] = 30;
    a[3] = 40;
    a[4] = 50;
 For (int i=0; i<5; i++)
   {
 system.out.Println (a[i]);
  }
 }
}
```

Advantage of Array :-

1. code optimization :-

It makes the code optimize. we can retrive ore short the data easily

2. Random access :-

we can gete any data located at any index Position.

Disadvantages of array :-

1. size limit :-
one can store only fixed size of elements in the array it doesnot grow it size at long run time to solve this problem, collection of framework is used in Java.

→ There are two types of array
1 single dimension of array
2. multi dimension of array

single dimension of array

Array can be constructed in Java by 3 steps
~~1st one is declaration~~
(1) Declaration
(2) Instantiation
(3) Initilization

sintax of Declare an array :-
datatype [ ] array Refvar ; or
data type [ ] array Refvar ;
data type array Refvar[ ]

```
class Test {
public static void main (String [ ] args)
{
int arr [ ] [ ] = {{1,2,3}, {2,4,5}, {4,4,5}};
    for(int i=0; i<3; i++) {
      for(int J=0; J<3; J++) {
system.out.println (arr[i][j] + " ");
    }
    system.out.println();
  }
}
```

[illegible]
[illegible]
[illegible]
[illegible]

[illegible]
[illegible]

[illegible]
[illegible]
[illegible]
[illegible]

[illegible]

[illegible]
[illegible]
[illegible]
[illegible]

[illegible]
[illegible]

[illegible]
[illegible]

[illegible]

[illegible]
[illegible]
[illegible]

[illegible]
[illegible]
[illegible]
[illegible]

[illegible]
[illegible]

[illegible]
[illegible]

[illegible]

[illegible] the order :

[illegible] → long → float → Double [illegible] to a long value and [illegible] value by using a [illegible]

```
[illegible]
{
[illegible] (String[] args)
{
[illegible] = 30;
[illegible] = [illegible];
```

Scanned with OKEN Scanner

```
double c=b
system.out.println(a);
 system.out.println(b);
system.out.println(c);
   }
    }
```

Explicit casting:

→ This type of casting involves assining a data type of higher range to lower range.

→ This in done manually as you need to do the casting using "( )" parath operat

If we fail to do the casting a compile time errore will be returned by the compiler.

~~[illegible]~~

Double → Float → long → Int → char → short → byte

```
        class main {
        psvm(string[] args)
         {
        double d=57.17;
      int i =(int)d;
    system.out.println(d);
    system.out.println(i);
         }
          }
```

Operators in Java:

→ Operators in Java is a symbol that is used to perform operation.

ex +, -, *, /, %.

→ There are may types of operator in Java which are given Java.

→ unary operator
→ arithmetical operator
→ relational operator
→ bitwise operator
→ logical operator
→ Ternary operator
→ Assignment operator

unary < postfix — expr++ , expr--
        prefix

unary < postfix → i++, j++, i--, j--
        prefix → ++i, ++j, --i, --j
                 ++expr, --expr

Arithmetic < multiplicative — *, /, %
             additive +, -

Shift - << >> >>>
Relation - < = > >=

bitwise < bitwise and → &
          bitwise exclusive → ^
          bitwise inclusive → |

logical < logical AND → &&
          logical OR → ||

Ternary = ternary — ?

Assignment - assignment =, += , -= , *= , /= ,
%= , &= , |= , <<= , >>=
>>>

Expression :-

→ Java expression consist of a variable, operator, literals and method.

→ Example :- int score
Score = 90; /
or
int score = 90;

→ Here, score = 90 is an expression that returns and integer.

→

* Double a=2.2, b=3.4, result;
result = a+b;

Here, a+b is an expression.

control flow statement :-

→ Java compiler executes the code from top to bottom. The statement is a code are execat according to the order in which they appear.

→ Here ever, Java Provides statements that can use to control the flow of Java code, such Statment are called control flows statment. It is one of the fundamental features of Java which Provides a smooth flow of program;

→ Java Provides 3 types control flows statment

1. Decision statment.
   → If statement
   → switch statment
2. Loop statement
   → do while Loop
   → while loop
   → for loop
   → for - each Loop

3. Jump statement

- break statement
- Continue statement

MODULE-3

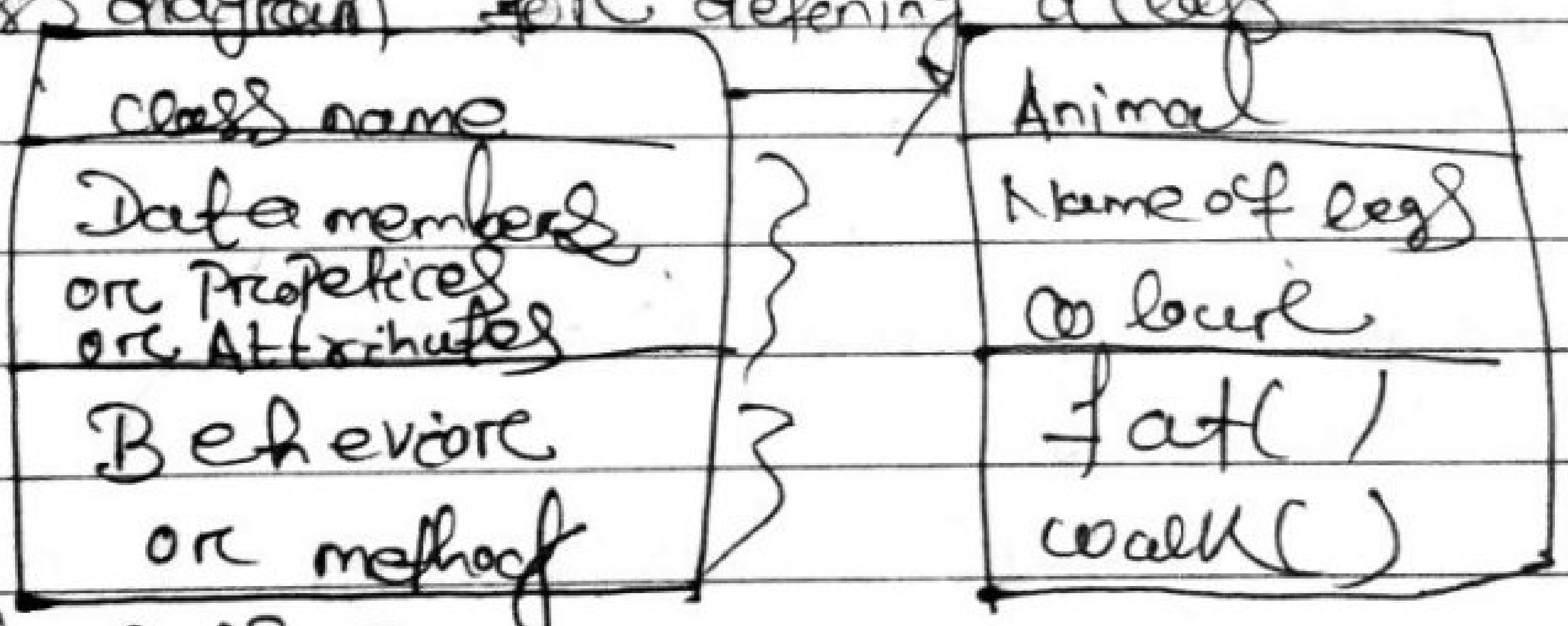
OBJECTS AND CLASSES:-

Concept and sintax of class:-

> A class is a way of binding the data and associated method in a single unit.

-> Any Java program if we want to ~~develop~~ develop than that should be develop than that should be developed with respective class only i.e without class there is no Java Program

> A class in Java can contain data member, method, constructor, block class diagram for defining a class



Sintax of class :-

```
class <class-name>
  {
   data member
   method member
  {
```

> Here class is a keyword which is used for developing or creating user defined datatypes

Concept of methods

> A method is a collection of statements that perfrom some specific task and return the result to the caller.

> A method can perfrom some specific task without

returning anything.

Syntax of a method:

The syntax to declare a method is return type methodname:

```
  // method body
}
```

For e.g:

```
 int addNumber(){
     //code
 }
```

Defining methods,

→ A method is a block of code which only runs when it called

→ You can passed the data known as parameters into a method.

→ method are used to perform certain actions, and they are also known as functions.

e.g:

```
        Public class main
        {
        Static void main method()
        }
```

Creating an object

→ The object is a basic building block of an OOPS language. In Java, we can't execute any program without creating an object.

→ There is a various way to create an object in Java that well discuss in this section.

→ Java provides three ways to create an object.

[illegible]}

Syntax of creating an object = new className();

```
public class Test
  {
   void show( )
   {
 Sopln (welcome to Java
 {
 psvm (String[] args)
  {
  Test t = new Test();
    t.show();
   }
  }
  }
```

Access of the members of a class from another class in Java

=> To access the members of a class from other class
-> first of all, import the class
-> create an object of that class
using this object access the members of that class

Suppose there is a class in a package called my package with a method named display()

```
     package mypackage
     public class Test {
      public void display(){
     System.out.println("Hello");
    }
    }
```

You can access it

```
     import mypackage
     public class myclass {
public static void main (String [] args)
```

Scanned with OKEN Scanner

Scanned with OKEN Scanner

-> using new Keyword
-> using clone ( ) method
-> using new instance ( ) method of the class.

Using new Keywords:

✓ Syntax of creating an object in Java:-

```
class Name object = new class name ( );
public class Test
{
void show( )
{ ~~system~~
System . out.println ("welcome to Aryan");
  }
public static void main(String [] args)
 {
 Test t, new test ( );
  t 1 , show( )
 }
 }
```

-> Access of the members of a class from another class in Java:-

-> To access the members of a class from other class

-> First of as import the class

-> create an object of that class using this object access the members of that class

* suppose there in a class in a package called my package with a method named display( )

```
package my Package ~~ci~~;
  public class Test
 {
  public void display( )
 {
  System .out. println("Hello");
}
}
}
for general it
normal and running;
public calling call;
}
public static void main(String args[])
  Test t = new Test();
    t.display();
}
}
}
class Simple
{
int a,b,total;
  void add()
{
total = a+b;
}
}
}
class Test
{
public static void main(String []args)
{
Simple s = new Simple();
        s.a=10;
         s.b=20;
        s.add();
System.out.println("Addition="+obj.total);
}
}
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
[illegible]
{
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
}
[illegible]
}
```

CLASSMATE
Date
Page

Acces specified inJava

acces ~~pecfi~~ Java allows to set the skop or accesibility or visibility of data member be it is a filed c class or method

Types of acces modifiere enJava

Java provides foure types of acces specified that we can use within classes and other entities, there.

(i) Default

whenevere the acces level is not specified, then it is as to be a defelt. The so skop of the defelt value is within the pakage.

(ii) Public

This is the most common access level and whenevere the Public access specified is used within and entitee that Partculare entite is accesable through out from within ~~to~~ out side the pakage out side the class.

(iii) Protected

The accessable has a sckop that is within the pavage a procted entitie is also accesable outside the pavage through out inherited class. if ~~and~~ you dont make the chiled class it cannot be. accesit from out side the pakage.

(iv) Acess control :

-> In Java accescontrol tells the Program howmuch acces a variable, class or method is given.

-> Access control is important because it affects visibility based on the different access control

-> when a variables or method access is not specified Public, or Privite it will have defelt visibility

-> Defelt visibility is Packed Private It is visible to all acces in the same pakage

Scanned with OKEN Scanner

⇒ It is used to declare members [illegible] to the [illegible] of an instance [illegible] in the base class, it can not be used by the other declared inside it can not be used in the [illegible] example of this class. In the [illegible] [illegible] It's [illegible]
: access control [illegible] not accessible private variable that is not accessible from [illegible] by the main method of the other class.

MODULE II (using Java String)

- String – mutable
- String buffer – immutable ([illegible])
- String builder –

String buffer : ~~[illegible]~~

```
~~class test~~ class test
{
public static void main(String [] args)
 {
  Test t1 = new Test("Hello");
     t1.append("Java");
 System.out.println(t1);
 }
}
```

STRING BUILDER :

```
public static class Test {
public static void main(String [] args)
{
String buffer = buffer new StringBuffer("[illegible]");
      sbuffer.append("Java");
system . out . print ln (barlofer);
}
}
```

13/12/24

METHODS AND MESSAGES:

-> A Java method is a collection of statements that are grouped togethore to perefrom an operation when you call the system. out printen ( ) method for. e.g the system actually executes several statements in order to display a message on the console.

-> Now we will learn how to create your own methods with or without return values, invoke a methods with or without parameters, and apply method abstraction in the program design

Creating method:-

Syntax:-

```
public static int method name (int a, int b) {
        // body
    }
```

Here

public, static modifier
int - return type
method name - name of the method
a, b - formal parameters
int a, int b - list of parameters

Messages:-

-> messaging is a method of communication between software components or applications

-> A messaging client send message to receive message from any other client

-> Each client Create with messaging agent that provides ~~fast~~ facility to send and receive messag

JMS:-

-) A Java messaging service is a Java API that allows applications to create send receive messages

-) JMS is a specification and JMS implementation as JMS Provider

-) Some JMS Providers are ActiveMQ, OpenMQ, oracleAQ

Parameter passing:-

There are different ways in which parameter data can be passed into and out of method and function. let us assume that a function B() is called from another function A(). In this case A is called the "caller function" and B is called the "called function or callee function". Also, the arguments which A sends to B are called actual arguments and parameters of B are called formal arguments.

Types of parameters:

Formal Parameter

A variable and its type as they appear in the proto type of function or method

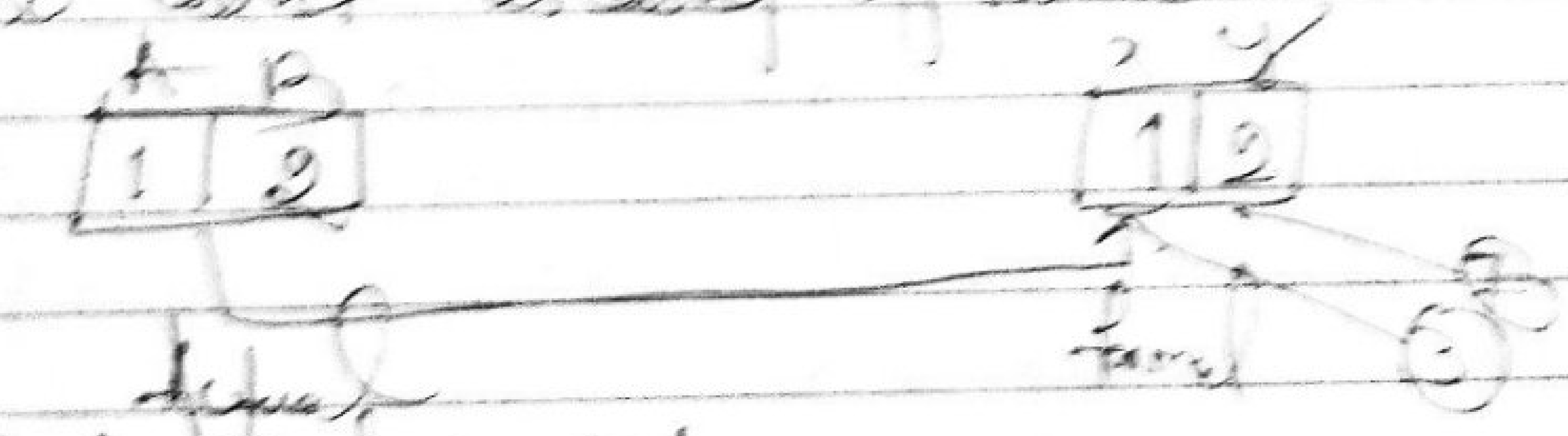
Syntax function name(data type variable name)

Actual Parameter:-
The variable or expression corresponding to a formal parameter that appears in the function or method in the calling environment.

func_name(variable name(s));

Important methods of parameter passing:-

1. pass by value:-
Changes made to formal parameter don't get transmitted back to the caller. Any modifications to the formal parameter variable inside the called function or method affect only the separate storage location and will not be reflected in the actual parameter in the calling environment. This method is also called as call by value.



2. Call by Reference:-
Changes made to formal parameter do get transmitted back to the caller through parameter passing. Any changes to the formal parameter are reflected in the actual parameter in the calling environment as formal parameter receives a reference to the actual data. This method is also called as call by reference.

Importing and Identifying data:-

-> Java object class is the super class of all the Java classes. All Java classes implements the object class by default

Scanned with OKEN Scanner

MODULE-5 INHERITANCE

Inheritance in Java (new [illegible])

-> Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of ~~a~~ a present object.

-> It is a important part of OPS (object oriented programming system).

-> Inheritance in Java is that you can create new classes that are build upon existing classes when you inherit from an existing class, you can reuse methods and fields of present class. moreover you can add new methods and fields in your current class also.

-> inheritance represents the IS-A relationship which is also known as parent-child relationship.

use of inheritance

-> For method overriding.

-> For code ~~[illegible]~~ Reusability.

method overriding:-

if a subclass (child class) has a same method as declared in parent class, it is known as method overriding ~~[illegible]~~

Reusability:-

As the same specifies, reusability is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. you can use the same field and methods already defined in the previous class.

Types of inheritance:-

-> on the basis of class, there can be three types of inheritance in Java

1. single inheritance
2. multilevel in-~~[illegible]~~ heritance
3. Hierarchical inheritance.

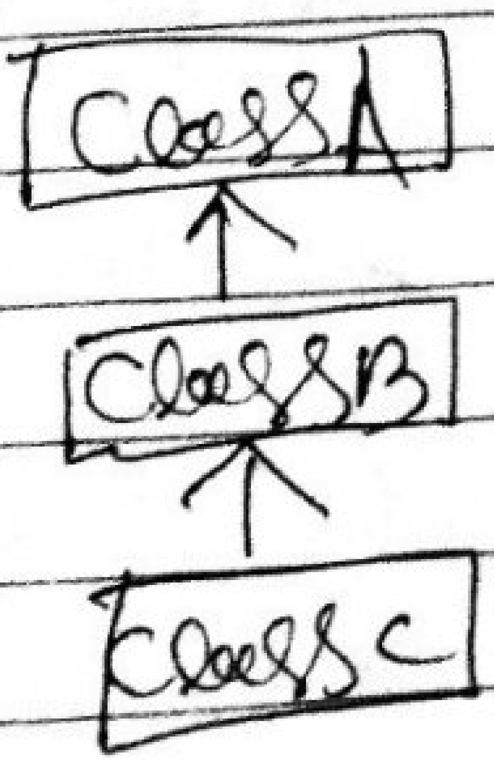
Scanned with OKEN Scanner

-> As you can see in the example give below Badf Dog class inherits the Dog class which again inherits the Animal class,

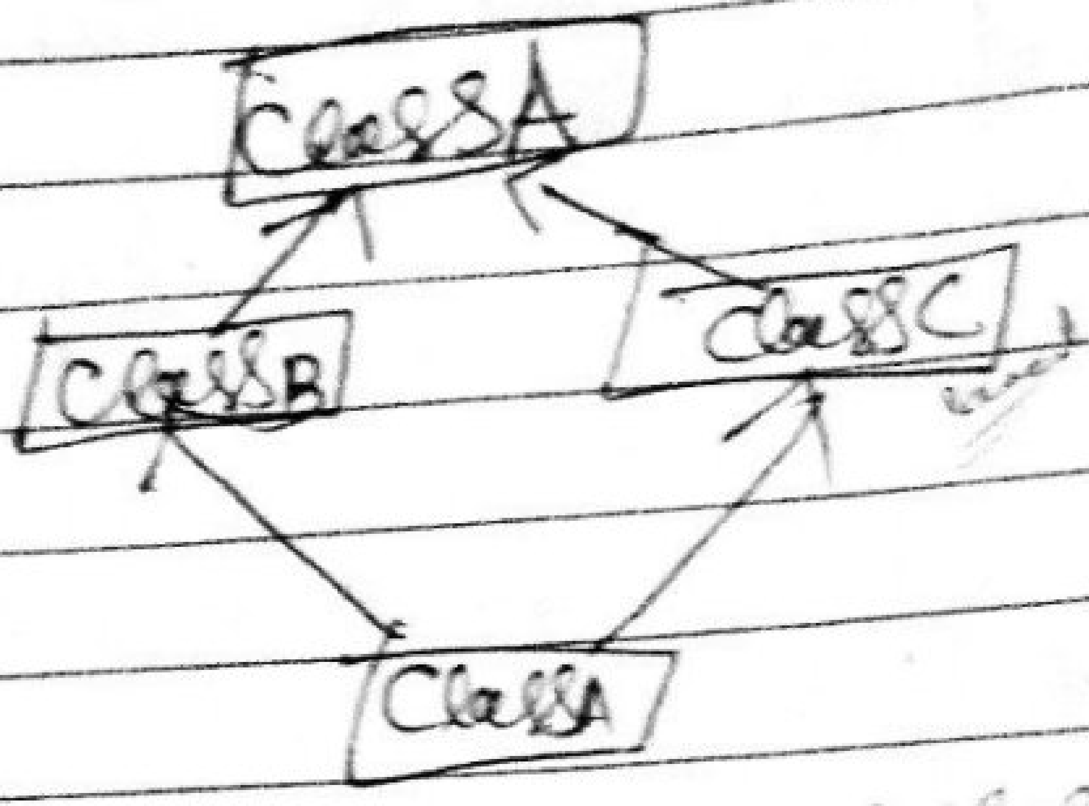
-> so there is a multilevel inharitance.

```
  Class Animal {
  void eatC()
  {
  Sopch ("eating");
 }
 }
Class Dog extends Animal {
  void bark( ){
 Sopch ("barking");
 }
 }
Class Badf Dog extends Dog {
void weep()
{
Sopch("weeping");
}
}
Class test inheritance? {
 P svm (String [] args)
  ~~Bad~~ Baby Dog d = new BadfDog();

 d. weep();
 d. bark();
 d. eat();
 }
 }
```



Hierarchical Inheritance:-



-> when two ore more classes inherits a single class, it is known as hierarechical inheritance

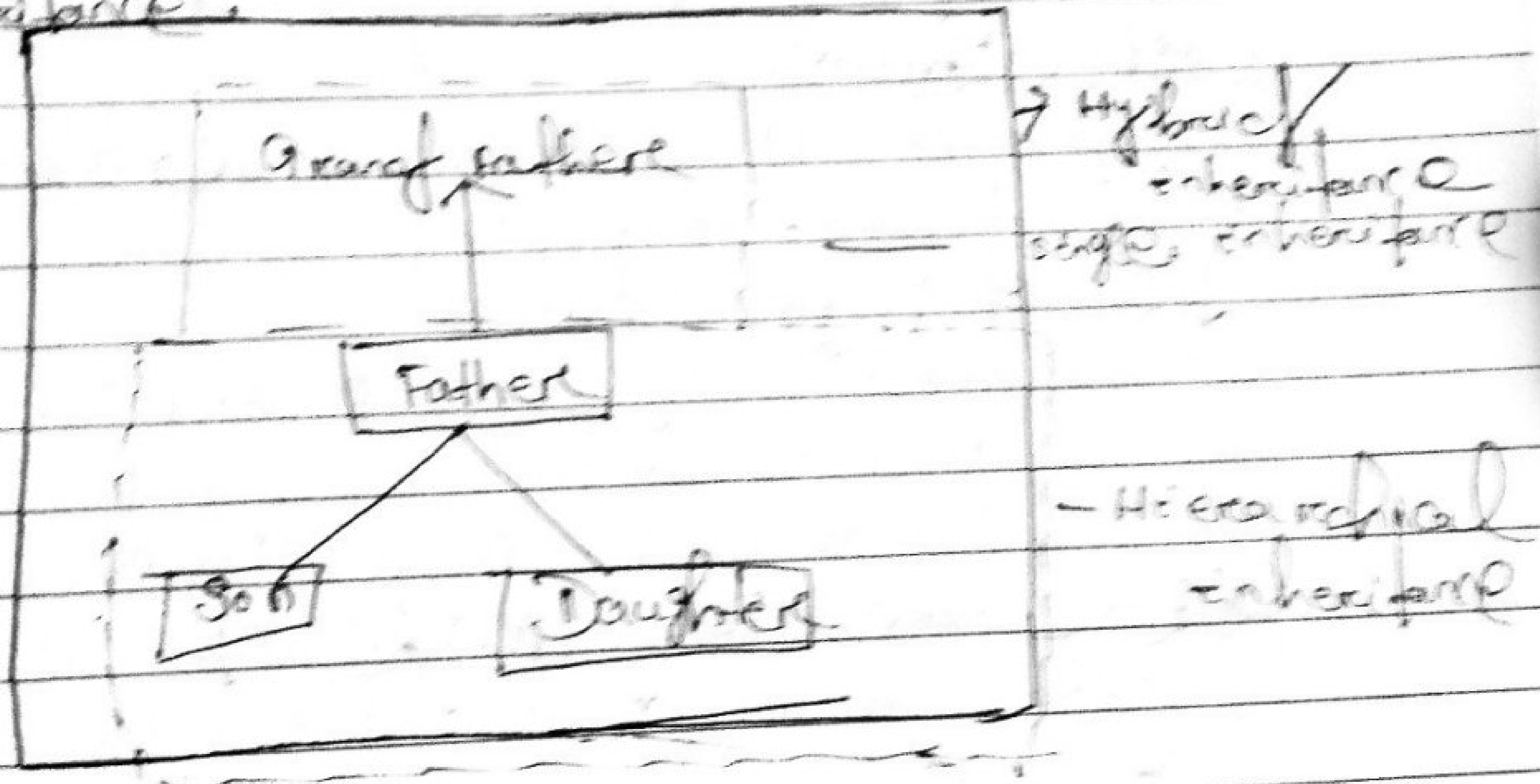
-> In the example given below, Dog and Cat classes inherits the animal class, so there is hierarchical inheritance.

```
  class Animal {
  void eat()
 }
 sopun("eating");
}
}
class Dog extends Animal {
  void bark()
 }
 sopan ("barking");
}
}
class cat extends Animal
 }
 void meiow( )
 }
 sopcn ("menaing"),
 }
}
```

```
class mul inheritance {
  P s v m (String[] args)
  [illegible]
  Calc c = new Calc();
      c.calc();
  }
}
```

Hybrid inheritance :-

Hybrid means, consist of more than one. Hybrid inheritance is the combination of two or more types of inheritance.



→ In the above figure, grand father is a super class. The father class inherits the properties of the grandfather class. since father and grand father represents single inheritance

→ Father, the father class in inherited by the ~~son~~ Son and Daughter class.

→ Thus, the father becomes the Parent class fore son and Daughter

→ These classes represent the hierachical inheritance.

→ ambiguity it doesn't be [illegible]

MODULE-6 Polymorphism

Polymorphism ÷
Polymorphism in Java is the ability of an object to have many ~~to~~ forms.

Types of Polymorphism ÷

There are two types of polymorphism in Java
1. Compile time polymorphism
2. Runtime polymorphism.

Polymorphism ÷
Polymorphism in Java is the ability of an object to have many forms.

Types of Polymorphism ÷
There are two types of Polymorphism in Java
1. Compile time Polymorphism
2. Runtime Polymorphism

Compile time Polymorphism ÷
A polymorphism which is exists at the time of compilation is called compile time or early binding or static polymorphism
e.g. – method overloading

Method overloading ÷

~~when~~ whenever a class contain more than one method have same and different type of parameters is called method over loading.

Syntax – return-type method name(

return -type method -name (param1 , param2)

```
class A
{
void add()
{
int a=10, b=20, c;
   c=a+b
   Sopln(c);
}
void add(int x, int y)
{
 int c;
 c = x+y;
  Sopn (c);
}
void add (int x, double y)
{
   double c;
    c = x+y;
    Sopn (c);
{
psvm (String [] args)
     A r= new A()
       r. add();
      r . add (100,200);
       r . add (50,45.32);
```

method same parameter different is overloading

parameter

Runtime polymorphism:

A polymorphism which exist at the time of execution of program is called run time polymorphism.

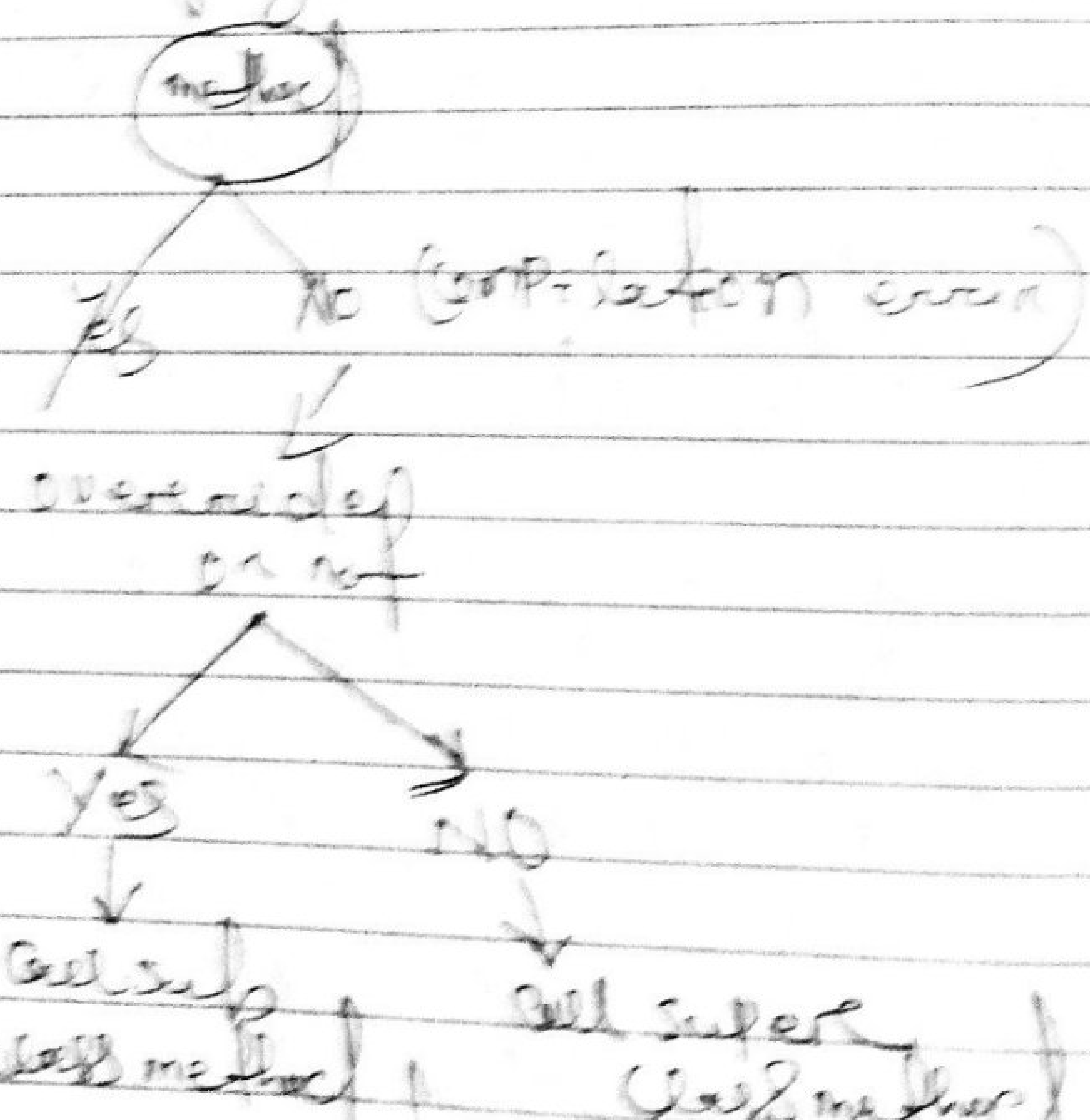
run time polymorphism.

e.g = method overriding.

Syntax:-

```
class A
{
void show()
{
}
}
class B extends A
{
void show()
{
}
}
}
```

method overriding rules:-



whenever we writing method in super and sub classes in such a way that method name and parameters must be same called method overriding

Packages : putting classes together

MODULE-7

Introduction :

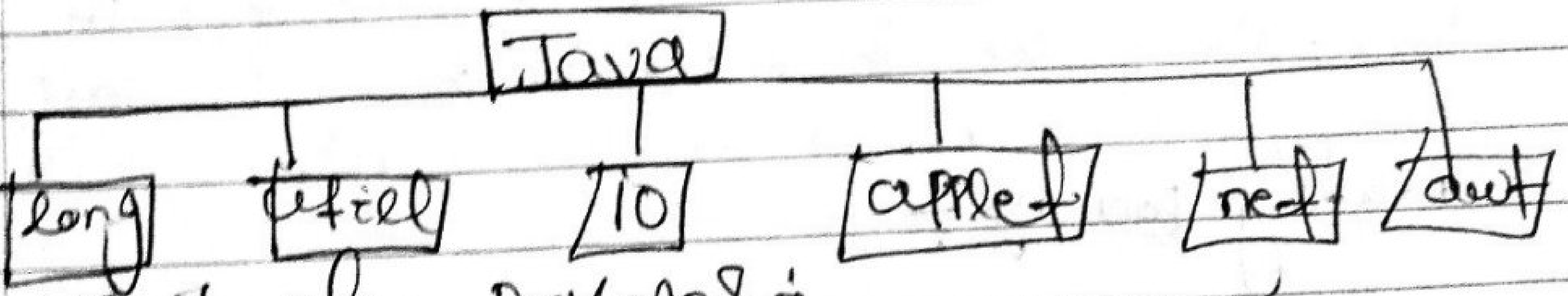
→ A Java Package is a group of similar types of classes, interfaces, and Sub-package.

→ package in Java can be categorized in two from, built-in package and user-defined package

→ There are many built in package such as Java, lang, awt, Javax, swing, net, io, util sql etc,

JAva API Packages :

Java API Application Program interface provides a large numbers of classes grouped into different packages according to to functionality most of the time we use the packages available with the Java API. following figure shown the system Packages that are Frequently used in the program.



using system packages :

Java language : language support classes. They include classes for primitive types. string, math, function, thread, exception.

Java utility : language utility classes such as vectors, hash table, random numbers, data etc.

Java . io : input/output support classes. They provide facilities for the input and output of data

Java.applet: classes for creating and implementing applets.

Java.net: classes for networking. They include classes for communicating with local computers as well as with internet servers.

Java.awt: set of classes for implementing graphical user interface. They include classes for windows, buttons, lists, menus and so on.

Naming Convention:

-> Java naming convention is a rule to follow as you decide what to name your identifiers such as class, package, variable, constant, method etc.

-> But, it is not forced to follow. So, it is known as convention not rule. These conventions are suggested by several Java communities such as Sun microsystems and Netscape.

-> All the classes, interfaces, packages, methods and fields of Java programming language are given according to the Java naming convention. If you fail to follow these conventions, it may generate confusion or erroneous code.

Creating package:

Creating a package is a simple task as follows

-> Choose the name of the package.

-> Include the package command as the first line of code in your source file.

-> The source file contains the classes, interfaces etc you want to include in the package

-> Compile to create the Java packages

PAKAGE IN JAVA

what is Package?

A Package is a group of similare types of classes, interfaces and sub-Packages, Package can be categorized in two from.

(i) Built-Package

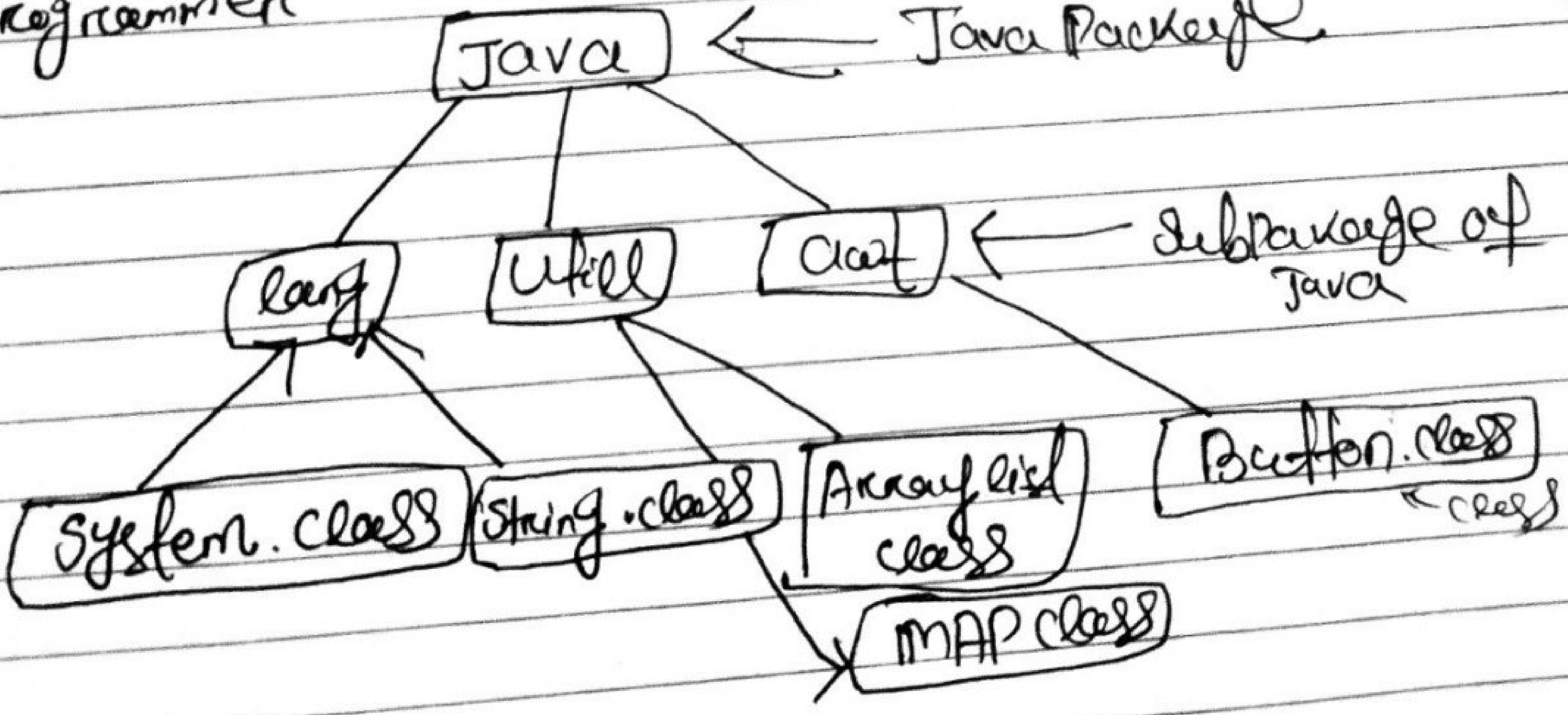
(ii) user-defined Package

Advantage of Package

- Package is used to categorize the classes and interface so that they can be easily maintained.
- Package Provides access Protection
- Package removes naming Collision.

Built-in Package:

Predefined Packages are those which are developed by Sun micro systems and supplied as a part of JDK (Java Development Kit) to simplify the task of Java Programmer



Usere definef Packages

A usere definef Package is on which is developef by Java Programmere to simplify the task of the Java Programmers to keep set of classes, interfaces and Sub Packages which are Commonly usef Any class ore interface is Commonly usef by many Java Programmers that class ore interface must be Placeef in packages "package" Keyword is usef to create a Package. whenever we create user defined Package we must use Package statement as a first executable statement.

EX To create a Package ÷

```
[illegible] pakege my pack
        Public class simple
                        public static voif main(string args[]
                        {
                            System.out.printen("welcome to Packages
                        }
```

To Compile. Java.c -d . simple. Java

To Run: Java my pack. simple.

Note - T-d is a switch that tells the Compiler where to put the class file. It represents destination. The (dot) Represents the Current folder and user may place the class file any directory by giving the directory name eg Java -d E:/ Java/ Java simple Jav

* How to access Package From anothere Package?

There are three ways to access the Package. 1. from outside the Package.

1. import Package-name*;
2. import Package-name. classname
3. Fully Qualified name.

Note:- if you use package.* then all the classes and interfaces of this package will be accessible but not of packages.

The import keyword is used to make the classes and interface of another package accessible to current package.

Ex of package name:-

```
Save by A.Java
package pack;
public class A
{
        public void msg()
        {
              System.out.println("Hello");
        }
}
Save by B.Java
package mypack;
import pack
class B
{
      public static void main(String args[])
      {
            A obj = new A();
            obj.msg();
      }
}
```

Output:

```
\sasuat\desKTOP\demo >javac -d .AJava
```

Example by package name: class-Name

```
save by A.Java
package pack;
public class A
{
        public void msg()
        {
                System.out.println("Hello");
}
// Save by B.Java
package mypack;
import pack.A;
class B
{
        public static void main(String args[])
         {
                A obJ = new A();
                obJ.msg();
        }
}
```

output:

```
C:\users\SASWAT\DESKTOP\demo>Javac -d. A.Java
 "          "          "        "  "  "
 "          "          "        "  "  "
Hello
```

If you use fully qualified name then only declared class of this package will be accessible. Now there is no need to import. But you need to use fully qualified name every time when you are accessing the class or interface.

EXAMPLE FULLY QUALIFIED NAM

```
// Save by A.Java
Package Pack
Public class A
{
        Public void msg()
          {
        System.out.println("Hello");
          }
}
// Save by B.Java
Package my Package;
class B
{
     Public Static void main(String args[])
      {
             Pack.A obJ = new pack.A(); // using fully
                                          Qualified nam
             obJ.msg();
      }
}
```

Java hiding classes:-

Hiding classes:

-> when we import a package with in a program only the classes declared as public in that package will be made accessible with in this program. In other words, the classes not declared as public in that package will not be accessible within this program.

-> we shall profitabey make use of the above fact. Some times, we may wish that certain classes in a package should not be made accessible to the importing package :- In such cases we need not declare the classes as public when we do so, these classes will be hidden from being accessed by the importing classes. Here an example

* my class Java

```
package mypack
public class myclass{
  public void display(){
  System.out.println("my class");
 }
}
class Defa class
 {
 public void display(){
   Sopln ("Defa ~~ca~~ class")
 }
}
}
```

Here the class Defa class which is not declared public is hidden from outside of the package my pack. This class can be seen and used only by other classes in the same package. Note that a ~~In~~ Java source file should contain only one public class and may include any number of non-public classes

```
    import mypack.*
  public class Javaapp{
  psvm (string [] args){
  Defa class da = new Defai class();
          da display();
    }
  }
```

Java Compiler would generate an error message for the Code Doctor class fc = new Doctor class( )",

Static import in Java:-

In Java, static import concept is introduced in... version. with the help of static import, we can access the static members of a class directly without class name or any object

e.g import static Java(any system class static import example{

```
        psvm (string) args)
    {
    out. printen ("Hello");
    out. Print ln ("Java);
}
}
```

MODULE-8

Java files AnD I/O:-

STREAM:

A stream is a sequence of data. In Java, a stream is composed of bytes. It's called stream of water that continues to flow

Reading and writing to files:-

Java File writer and file Reader classes are used to write and read data from text from text files. (they are character stream classes) It is recomm-ended not to use the file input stream and file output stream classes if you have to read and write any textual information as these are byte stream classes.

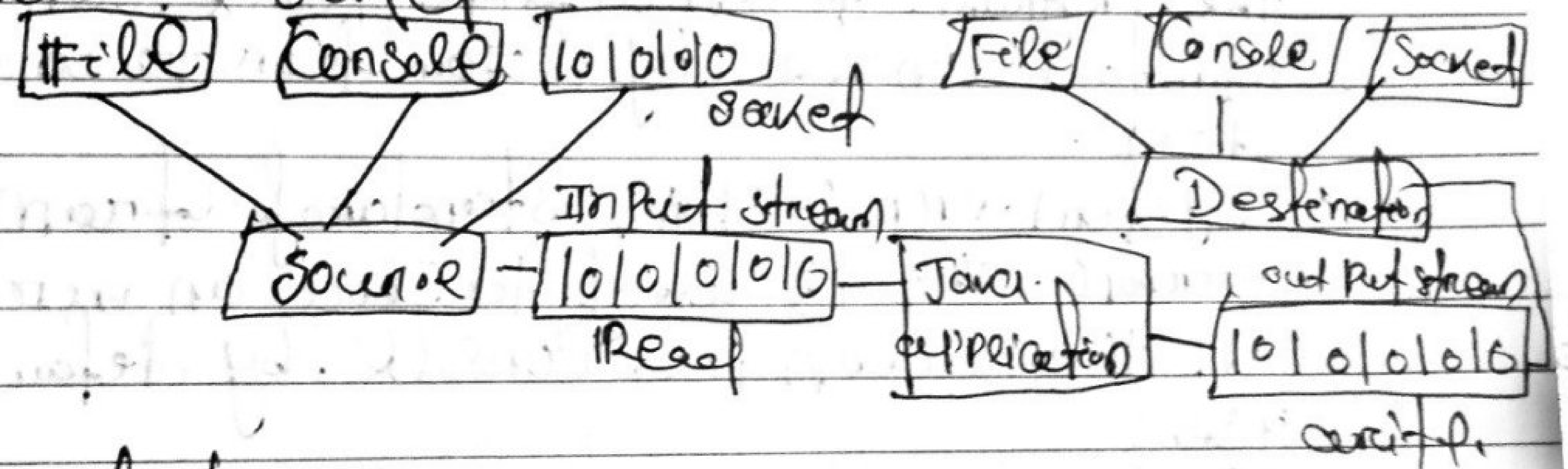
File Reader:

It is useful to read data in the from of characters from a "text" file

output stream/input stream:

Output stream:

Java application uses an output stream to write data to a destination. It may be a file, an array, Peripheral device or socket



• Input stream:

Java application uses an input stream to read data from a source. It may be a file, an array, Peripheral device, and socket.

Opening and Closing streams:

~~Streams have a base~~

streams have a base stream close() method and implement Auto close, and but nearly all the stream instances donot actually need to be closed after use. Generally, only streams whose source is an I/O channels (such as those retoring by files lines(Path, charset) will require losing)

Predefined streams :-

Three Predefined streams are standard streams are available in Java lang.system class

system in → This is the standard stream for input data. This stream is used for reading data for the program from the keyboard by ~~da~~ default

system.out → This standard stream for output. This stream is used for writing data for the program to an output device, such as a monitor console by default or to some specified file.

System.err :- This standard stream for error. This is used to show an error message on the screen i.e console by default for the users.

File handling methods :-

~~The File cla~~ The file class have many useful methods for creating and getting information about files.

Methods	Type	Description
canRead()	Boolean	Tests whether the file is readable or not
~~canwrite()~~	~~Boolean~~	
canwrite()	Boolean	Tests whether the file is writable or not

createNewFile()	Boolean	Creates an empty file
delete()	Boolean	Deletes a file
exists()	Boolean	Tests whether the file exists
getName()	String	Returns the name of the file