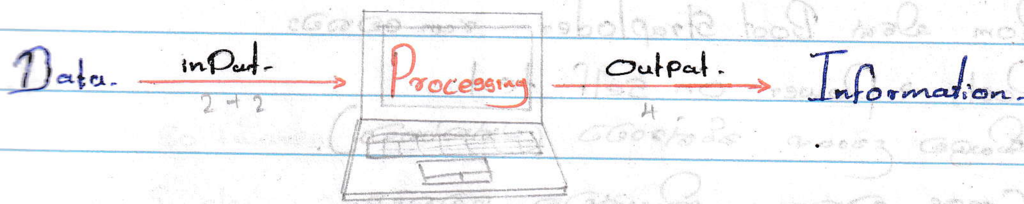
No:

Date: 20/02/2024

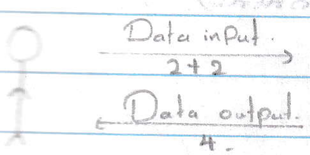
The Computer.

What is the Computer.

* The Computer is electric power using device and when we input the data. It Can Process and give output for us.

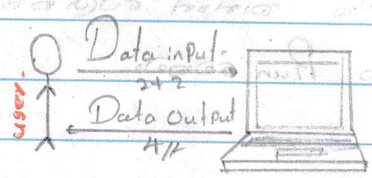


Human vs Computer.



(නිරවද්‍යතාව)

* Accuracy. : ↓
* Speed. : ↓
* Tired. : Easily tired.



* Accuracy : ↑
* Speed. : ↑
* Tired. : No Tired.

Hardwares.

* Motherboard.
* Cpu / Processor.
* Ram
* Hard disk. (Hdd, SSd, M.2).
* Power Supply
* Cooling fan.
* Gpu / VGA
* Battery

Softwars.

System Softwares
- windows
- mac os
- Android

Application Softwares.
- Chrome

Utility Softwares
- Virus guards.

No :

Date:

⊙ How to Computer start., what happen.? / what is the Co Boot prosess?

1) Power Supply එකට Current එක යනවා.
2) Cpu
3) Bios Run වෙලා -
   Rom එකේ Boot Straploder run වෙනවා
4) Post → Power on self test.
   (සියලුම උපාංග නිවැරදිව (චෙක්කරනවා.)).
5). Cmos මතකය ක්‍රියාත්මක වෙනවා.
6) MBR Run වෙනවා.
   MBR → Master Boot Recorder.
   (Os එක ගබඩා වෙලා තියෙන තැන MBR එකේ තියෙනවා.).
7). Os එක Run වෙනවා.

INCOMPLETED
Date: 2024/02/27
Signature:

⊙ What is a hardware. :

Computer hard-ware : is a physical device Of C... that what we Can see and touch.

ex : • Monitor
- Cpu → Central Processing unit.
- mouse
- Joystick

Computer Hardwares.

No : _____ Date: __/__/____

⊙ What is a Software. ? -

In a Computer System, the Software is basically a set of instructions and Commands that tell a Computer what to do. In Other words, the Software is a Computer program that provides a set of instructions to execute a user's Commands and tell the Computer what to do.

Softwares.

→ System Softwares. ⟶

* System Software is software that provides a platform for other Softwares.

ex → * windows * Ubantu.
* Mac * Linux.
* Android.

→ Application Softwares. ⟶

* The Application Softwares are the Software design to handle spaecific tasks for users.

ex → * Chrome.
* brave.
* Ms office.

→ Utality Softwares. ⟶.

*. Utality Software is a type of the Software that is designed to help users manage, maintain and optimize their Computer Systems.

ex → * Antivirus.
* file management System
* Disk Clean up tool

No :

Date: __/__/____

⊙ What is a firmware?

* firmware is a small software that is written to hardware devices in non-volatile memory.
* we cant erase the firmware softwares I was written by the manufacturers when it manufacturing.

Bios	Cmos.
⊙ Non-Volatile.	⊙ volatile.
⊙ Basic input output System is a long form.	⊙ Complementary Metal oxide semiconductor is the long form
⊙ Initialize hardware while booting up the Computer and Provide runtime Services for Os. and Programs.	⊙ A special memory Chip in the mother board that stores and holds the bios Configuration Settings.

⊙ Linux Operating System.

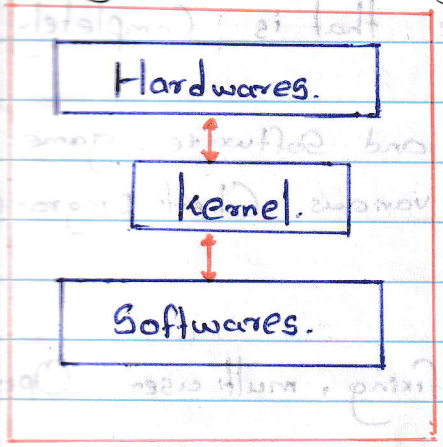
What is the linux Os?

* The Linux kernal is a heart Of the linux Operating System.
* It is an Open Source. (fully free).
* Linux is Created by Linus Torvalds. in 1991
* Linux Source Code is Open for everyone to explore and ~~modify~~ modify.

No: _____ Date: __/__/____

What is a Kernel ?

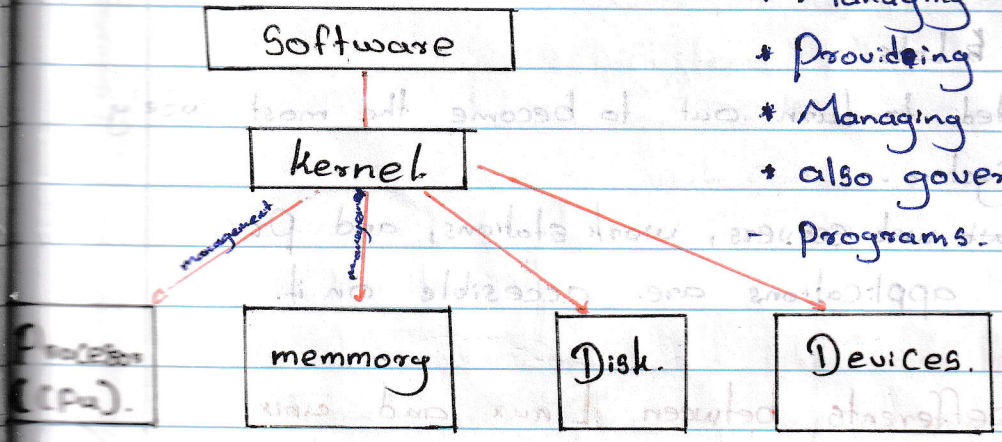
An Operating System is a Software System that manage the Computer that provides some services for Computer Programs and manages Computer hardware and Software.



Basically, it is a Communication or resource allocation between Computer and hardware and applications.

~~It~~ The kernel provides some services. they are.,

* Managing input and output devices.
* Managing file systems
* Providing UI (user Interface).
* Managing Computer memory
* also govems and executes all the programs.



⊙ what is a Linux distribution.?

(Linux ...)

A Complete linux System Package Called a distributions many linux distributions are available to meet just about any Computing requirement you Could have. Most distributions are Customized for a Specific user group, Such as business users. Multimedia enthusiasts, Software developers, or average home users.

No: _____ Date: 27/02/2024

What is the Linux ?

* Linux is an Operating System that was developed by Linus ~~Tra~~ Torvalds in 1991. The name "Linux" Originates from the Linux kernel.

* It is an Open-Source software that is Completely free to use.

* It used for Computer hardware and Software, game development., mainframes, etc. It can run various Client programs.

What is the unix ?

* Unix is a protable, multitasking, bug-fixing, multi user Operating System devoleped by AT & T

* It started as a one man venture under the initiative of Ken Thompson of Bell & Labs.

* It ~~use~~ proceeded to turn out to become the most widely use & Operating System.

* It is used in ~~web~~ web servers, work stations, and pcs.

* Many business applications are accesible on it.

What are the defferents between Linux and unix ?

Linux	Unix.
⊙ Linux was developed in the ~~1990s~~ 1991 by Linus Tovalds as a free and open-source alternative to unix (Bad).	⊙ Unix was developed in the 1970s at Bell Labs.
⊙ Linux is a open-source and a large number of	⊙ unix was developed by AT&T Labs, different Commercial vendors,

No: _____ Date: __/__/____

⊙ Linux, on the other hand, open-source software can be freely without any licensing fee.

⊙ Unix is a proprietary operating system. we need license to use.

Both have a similar design.
But
Unix kernel is lager and
more Complex than
the Linux kernel.

Both are used in Severs,
work stations, emberded
Systems. But
Linux ~~dometing~~ dominates in Open source
and web Server markets.

What is linux directory and what are the it ~~fo~~ purposes.

In the Linux Operating system everything is a file even directories are file.

Types of files in Linux System.

→ General files. -

It also called as Ordinary files. It may be an image, vedio, Program, or Simple text file. These types of files can be in ASCII or Binary formal. It is the most Commonly used file system in the linux System.

→ Directory files. -

These types of files are warehouse for

No :

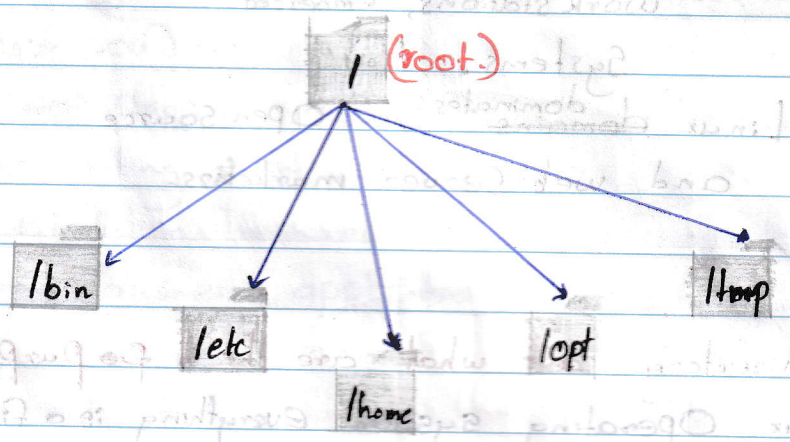
Date: __/__/____

within a derectory.

→ **Device files.-**

In a windows. like operating system, devices like CD-Rom, and hard drivers are as drive letters like "F", "G", "H" wheears in Linux system devices are represented as files. As fer example.- /dev., /

⊙ Linux Operating system files are Stored in a tree like Structure star ting with the root directory



Directories.	Description.
/bin.	binary or executable Programs
/etc	System Configuration files.
/home	home directory. It is the currunt default directry
/opt	Optional or #third party software
/temp	tempory space. typicaly Cleared on re boot
/usr	user. related Programs.
/Var.	Log files.

No: _____ Date: __/__/____

What is the terminal.?

Terminal. → Ctrl + alt + T

The terminal is a program that provides the user with a simple Command line interface and profoms the following 2 tasks.

* Input takes from the user in the form of Commands.
* Displays output on the screen.

Linux Basic Commands.

* Sudo. – Super User do.
* Clear. → Terminal will be Clear.
* Pwd. → get Current path form
* Cd. → Change directory.
* ls. → list.
* Mkdir → Make directory
* touch. → Create a file.
* cp. → Coppy source file.
* cp -R → Copy the entire directory.
* mv. → move and Rename.
* rmdir → for empty directory.
* rm -r → Remove.
* touch. → Create a file
* locate -i → Search anything.

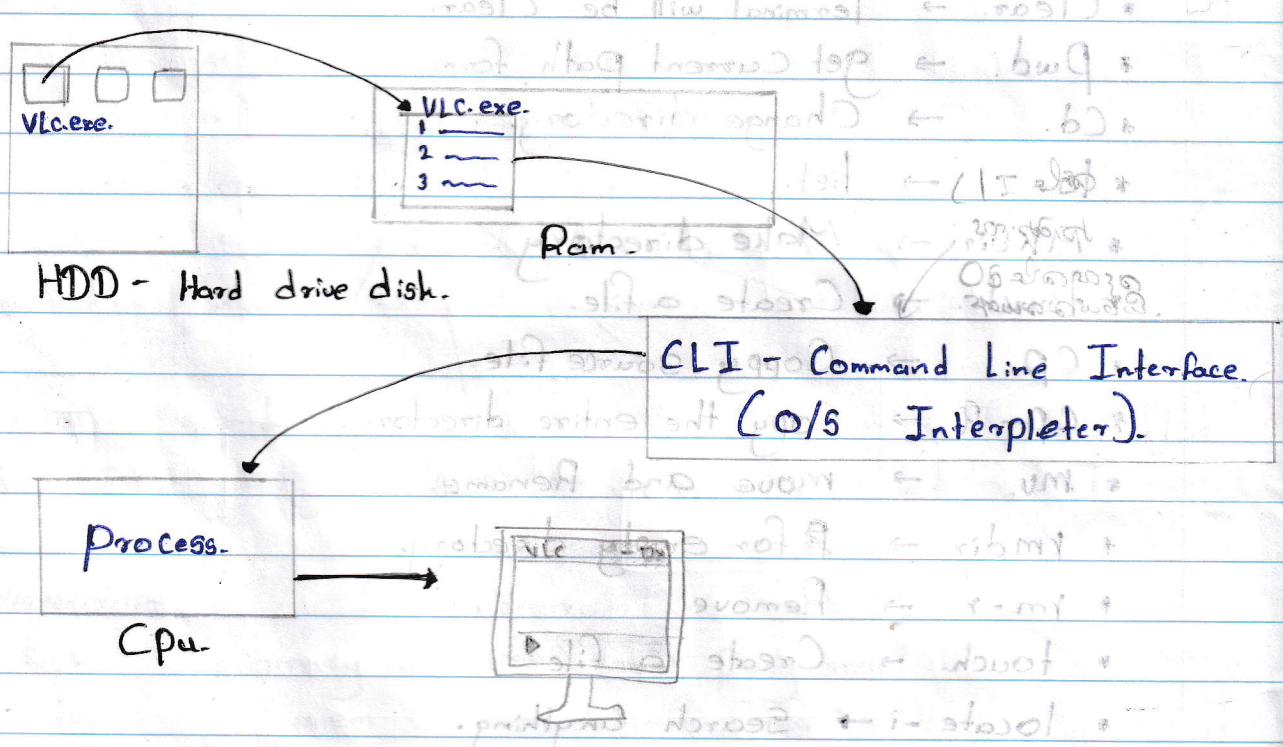
No : _____ Date: 04 / 09 / 2024.

* What is a Computer Program?

A Computer program is a set of Commands. / Instructions with Corresponding order for the profome a specific task.

- ~~A programm~~ A Computer program in its human - readable form it Called Source Code. Source code needs another Computer program to execute because Computer only Can understand only Mashine Language. Compiler is used to translate Souce Code to mashine Language all at Once.

* How to run a Computer Program?



Ex: VLC.exe ⟶ file.

* VLC.exe file is on ~~the~~ volatile memmory (HDD) when it open It takes memmory space from the ram ~~Ram is a~~ The ram is non-Volatile memmory.
When vlc runing there instructions and Commands are loded to cpu by using Command Line Interface after that we Can see vlc application Interface on our Screen

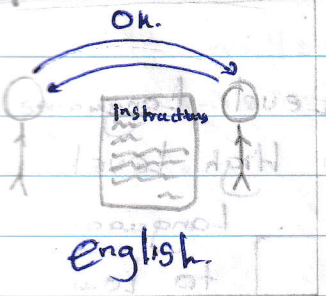
No: Date: __/__/____

• Who is the Software Engineer ?

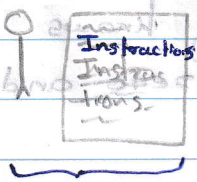
Software Engineering is the process of designing, developing, testing and maintaining Software. Software Engeneers give Solutions for real world problems.

• How to make Computer Program ?

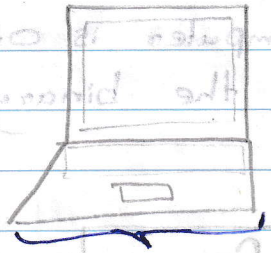
Computer Software consists of a series of instructions in a programming language; the programmer Compiles those Statements into a form that a Computer processor can understand. Building Software requries knowlage of Computer Languages, Syntax and logic to Complete the process form start to finish. In addition to the technical knowlage required, a programmer must be familiar with the Specialized Software tool needed in the form of an edifor, a Compiler and a debugger.



(Human - Language)
Can understand.

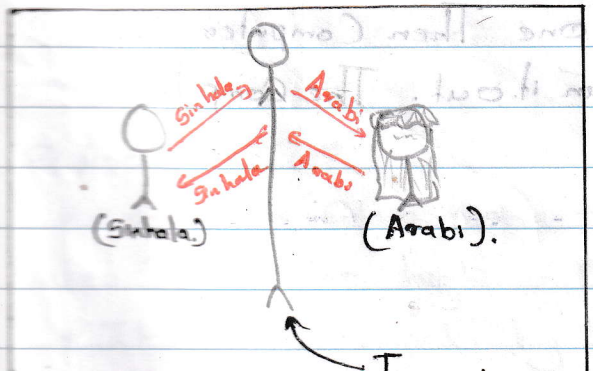


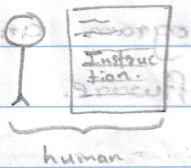
useing
Human - Language.
• Aa, Bb. Cc...



Computer useing
the mashine Language.
/ Binary Language.
• 1, 0

• Then why we need Programing Language ?

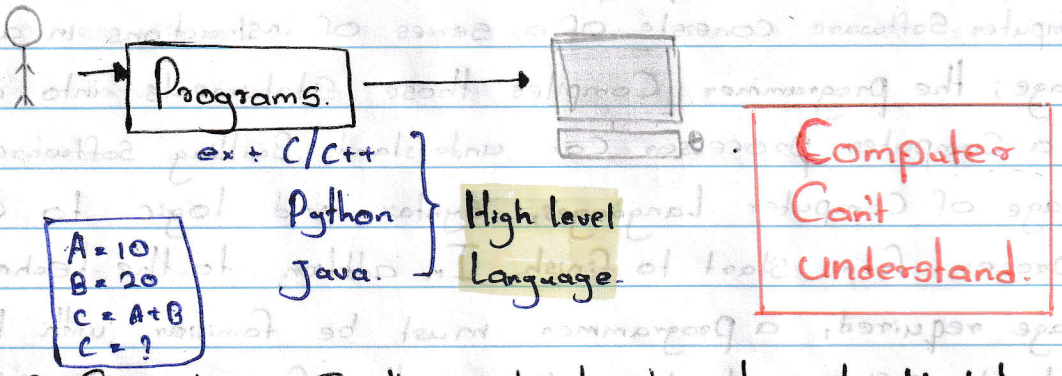




No : _____ Date: __/__/____

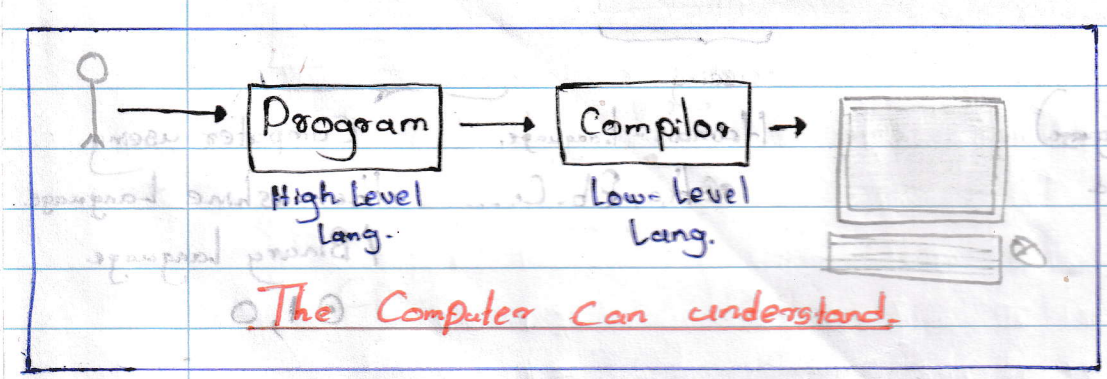
Compute Language also have rules to form statment Or instructions. The set of the instructions Called the Program.

Human Can't derect Connect with Computer. Human all users Can Communicate writing Programs.



⊙ Computer Can't understand derect. High Level Language. Because It is English.

Computer is only knows the Low Level Languages. as the binary zero and one. So High Level Language to Low Level Langage Convert. for this we need translator



The translator use hear is a Compiler. the Compiler Convert a program that uses write On Compater undestanable. forms zero and one Then Computer Can execute the program and run it.out. The Compiler is a System Software.

No : _____ Date: __/__/____

* Programing Languages and features.

* Python.
* C
* C++
* C#
* Java.
* Dart.
* Java script.
* Php
* Kotlin.
* Shift.
* go.
* Ruby
* PasCal.

⊙ About python.

* Python is a high-level genaral purpose Programming Language.
* Guido van Rossum is the father of python.
* frist appeared 20th February 1991; 33 years ago.
* Python is a multi paradigm Program language. Object Oriented Programming and Structured Programming are fully supported.

⊙ About "C"

* C is a genaral-~~pou~~ purpose Programming Language Created by Dennis Ritchie at the Bell Laboratories in 1972.
* It is very popular language, despite being old The main reson for its property is because it is a fundamental language in the field of Computer science.
* C is strongly associated with unix, as it was developed to write the unix operating system.

⊙ About "C++"

* C++ is a ~~Class~~ Cross platform language that Can be used to Create high-proformence applications.
* It was devoloped by Bjarne Stroustrup as an extention

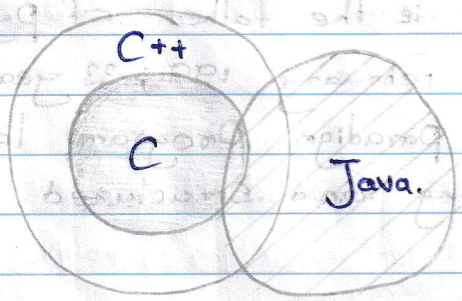
No: _____ Date: __/__/____

⊙ About Java.

* Java is a Programm language. Created in 1995.
* It's owed by Oracle and more than 3 billion devices run Java.

⊙ It is used for :-

* Mobile applications (specialy for android apps)
* Desktop applications.
* web applications.
* web Servers and application Servers.
* games.
* Database Connection and etc.



⊙ Java versions.

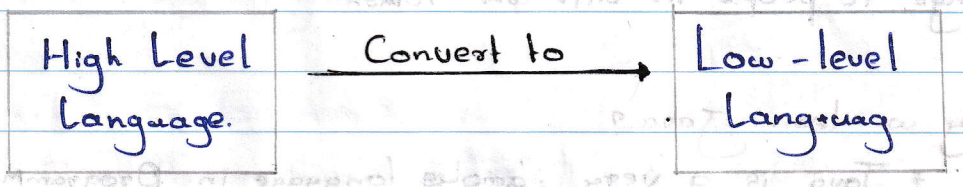
JDK 1.0	Java SE 14
JDK 1.1	Java SE 15
JDk 1.2	Java SE 16
JDk 1.3	Java SE 17 (LTS)
JDk 1.4	Java SE 18
JDK 5.0	Java SE 19
~~JDk~~ Java SE 6	Java SE 20
Java SE 7	Java SE 21 (LTS)
Java SE 8 (LTS)	Long term Support.
Java SE 9	
Java SE 10	(every 6 months
Java SE 11 (LTS)	Java relese
Java SE 12	

No: _____ Date: __/__/____

What are the common fetures in Programming Language?

⊙ Compiler. →

In Computing, a Compiler is a Computer program that transforms source code from a high level language ~~to~~ into a low level programming language.



human can understand.
ex:- Sinhala, english...

Computer can understand.
ex :- mashine Lang.
Binary Lang.

⊙ Interpliter →

All high level languages need to be Converted to mashine code so that the Computer can understand the program after taking the required inputs. The software by ~~with~~ which the Conversion of the high level introductions is proformed line-by-line to mashine level language, other Compiler and assembler is known as Interpliter.

⊙ Pre-defined (API) →

API full form is an Application Programming Interface that is a Collection of Communication protocals and subroutines used by various programs to communicate between them. A programmer can make you various API tools make their Programs easier and Simpler. Also, an API facilitates programmers with an efficient way to develop their software programs. Thus, api meaning is when an API helps two programs. ~~or~~ Or applications to Communicate with each other providing them with the necessary tools and funtions.

No: _____ Date: __/__/____

⊙ Documentation. →

Software documentation is written text or illustration that accompanies Computer Software or is embedded in the source Code. The documentation either explains how the Software ~~opp~~ operates or how to use it. and may mean different things to people in different roles.

⊛ Why we learn Java.?

* Java is a very popular language in Programming Languages.
* It ~~must be~~ is a. Simple, Object Oriented and familiar.

Object Oriented mean Progomming Paradigm based on the Consept of Objects.

* It is multi treding Programming Language.
* Java is platform independent. Java byte Code run on all the operating systems.
* High proformance.
* High Security.

Java.pr.①

~~Extra additional Information~~

⊛ How to run Java Program.

* frist you need to open your terminal. and request a Source Code file.

gedit Example.java.

gedit mean text editor in ubanta os as like the note pad on windows

Example.java. → after that when open the text editor their open the Java file Renamed with "~~Exte~~ Example"

* Then we need to write the Source Corde and save it and close it.

```
Public Class Example {
    Public static void main (String args [])
```

No: _____ Date: __/__/____

- after that we need to Compile our Source Code. to byte Code when we It done we can see ".class" file Create

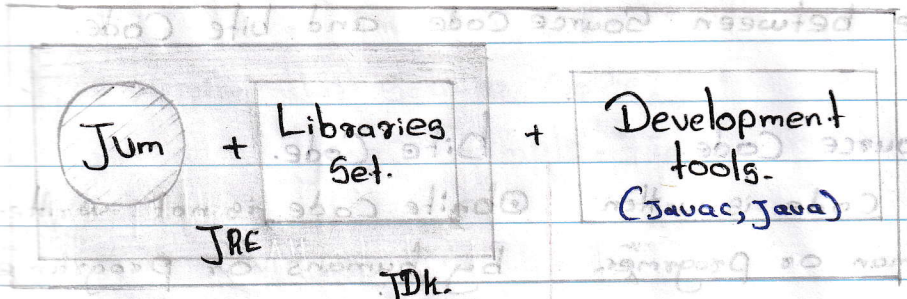
  javac Example.java.

- finaly we need to Run byte Code and see Output of our Program.

  ~~Java~~ java Example.

INCOMPLETED
Date: 14/3/2024
Signature:

① JDK. – Java Development kit.



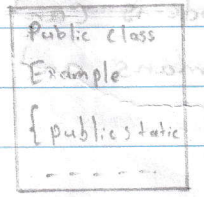
JDK → Java Development kit.

JRE → Java Runtime Environment.

Jvm → Java vertual mashine.

② How to Java Code Runs.?

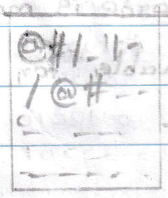
High-Level Language.



Example.java
(Sorce Code)

Javac
(compiler)

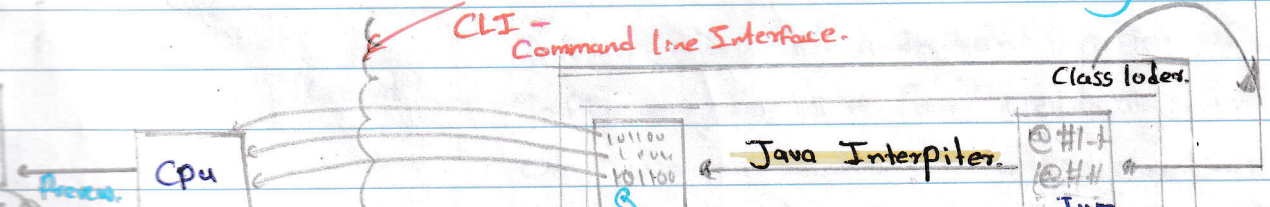
Low Level Code.



Example.class.
(Byte code / classfile / Inter-mediat Code.)

Java
(Run)

CLI – Command line Interface.



No : _____ Date: __/__/____

* **Source Code →**

* Collection of Computer instructions written useing some Human readable Computer language

* **Byte Code / Intermediate Code / Class code →**

* Byte Code can be defined as an intermediate Code genarated by the Compiler ~~Co~~ after the Compilation of Souce Code. (Java program).

* This intermediate Code ~~genarated by the~~ makes Java a platform - independent Language.

* Difference between Source Code and bite Code.

Source Code.	bite Code.
⊙ Source Code is written by human or programmer.	⊙ Bite Code is not written by humans or programmers.
⊙ High level programming Languages.	⊙ Bite Code is an intermediat Code between Source Code and machine Code.
⊙ Source Corde Cannot understand derectly by machine.	⊙ Bite Code is executable by Jvm (Java verchaal machine).
⊙ Its High Level Code.	⊙ It's Intermediate Code.
⊙ Source Corde is more understanable for humans.	⊙ Byte code is cant understand for humans and Computers.

No: _____ Date: __/__/____

How to Run Java Programm

- Open terminal and request a Source Code file.

```
gedit Example.java
```

- write the Sorce Code Save and Close it.

```
Public Class Example {
    Public Static Void main (String args [I]) {
        System.Out.println ("Hello world!");
    }
}
```

- Compile the Source Code.

```
Javac Example.java
```

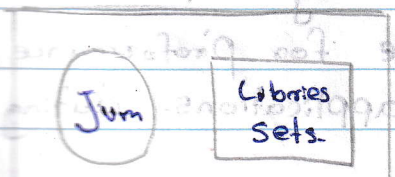
- Run byte code file.

```
java Example
```

JDk. – Java Development kit.

Java development kit is a distribution of Java Technology by ~~oracale~~ Oracle. Corporation. It is use fore developing Java applications. its one of the tree Core technology Packages used in Java progrramming along with the JvM, the JRE and the JDK include and interpleter / ~~border~~ loder, Javac Compiler and achiver (Jar) and Other tools need in Java.

- JRE → Java Run Time environment.



Java runtime environment is set of Software tools ● for development of Java applications. It is combines

No : _____ Date: __/__/____

* Jvm - Java virtual machine

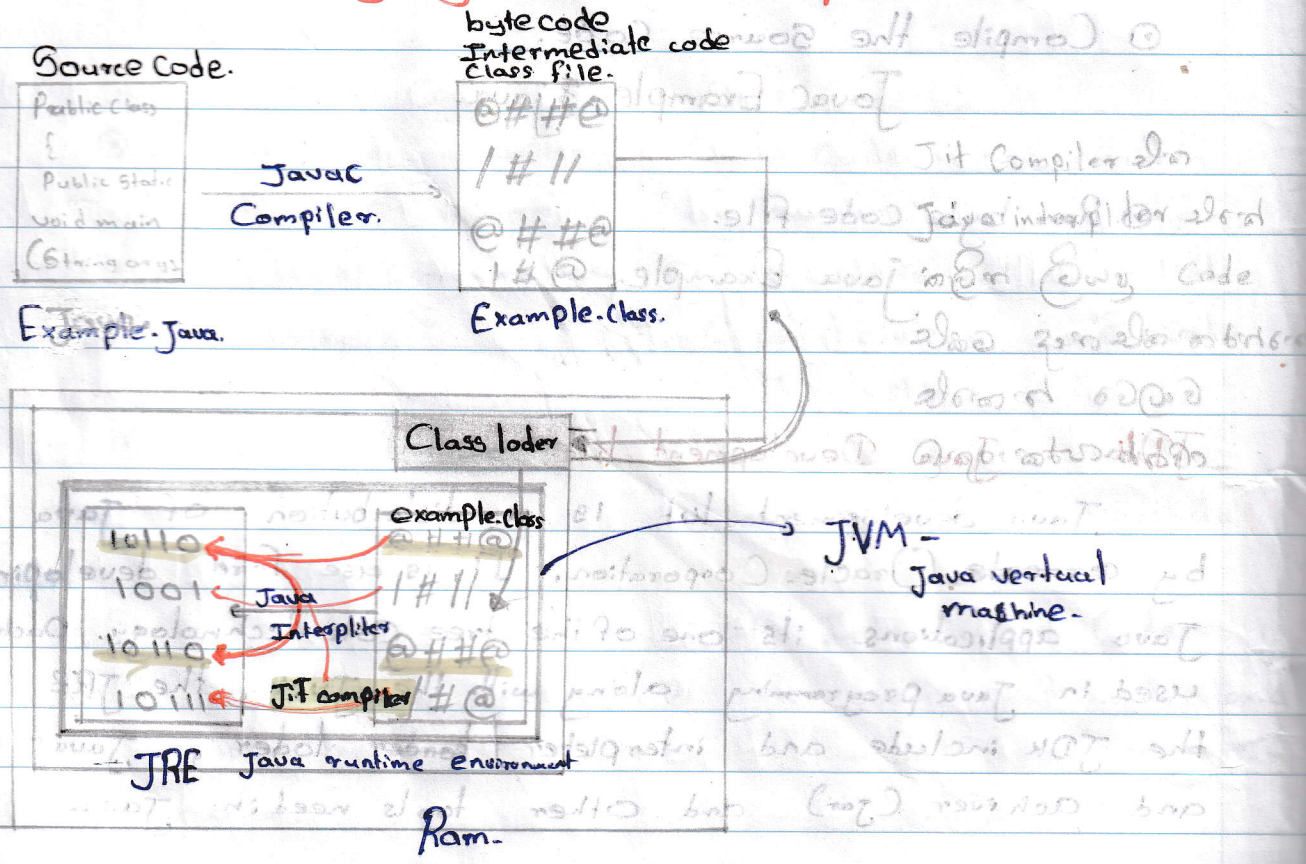
Java virtual machine is an abstract machine It is specification that provides run time environment In witch byte code can be executed.

* Jvms are platform dependent. (available for many hardware and software platforms).

2024/03/14.

* JIT Compiler → Just in time Compiler

what is Done by Just in time Compiler (JIT compiler)



The JIT Compiler (Just in time Compiler) is an essantial part of the Java Runtime environment. that is responsible for preformance optimization of Java based applications. during the run time.

No: _____ Date: __/__/____

at run time and Compile suitable byte code sequences into native machine code. while ~~use~~ using a JIT Compiler, The hardware is able to execute the native code, as Compared to having the Jvm Interpret. The same sequences of byte code repeatedly and incurring over head for the translation process. This subsequently leads to proformance gains in the execution speed, unless the Compiled methods are executed less frequently.

Fedures of Java.

Platform independent. →

Java is a platform - independent Programming Language. Java didn't requie the entire code to be rewritten for all the different platforms. It support platform independence. useing Java byte code and Java virtual machine Java Compiler Javac Converts the program code in to byte code. This Bite code is platform - independent and can run on any JVM on any ~~Platform.~~ opperating System.

Object Oriented Program →

Java is a purely object oriented programming Language which means that everything in Java is an object. and Supports Concepts like encapsulation, inheritance and Polymorphism.

~~write~~ WORA - (Write Once, Run Anyware).

Java code can run on any platform that supports Java without need for recompilation., making it highly portable.

No : _____ Date: __/__/____

⊛ Secure. →

Java has built in security features., Such as byte code verification and a secure runtime environment that make it a ~~Sec~~ Secure platform developing applications.

⊛ Protable. →

As we know Java code written on one machine can be run on another machine. The platform independent feture of Java in witch it is platform independent byte code can be taken to any platform for execution ∴ makes Java protable.

⊛ High proformance. →

Java architecture is defined in such a way that it reduces overhead during the runtime and and sometimes Java uses Just in time ~~Com~~ (JIT) Compiler. where the Compiler Compiles Code on demand basics where it only compiles these methods that are called making applications to faster.

⊛ Multithreading. →

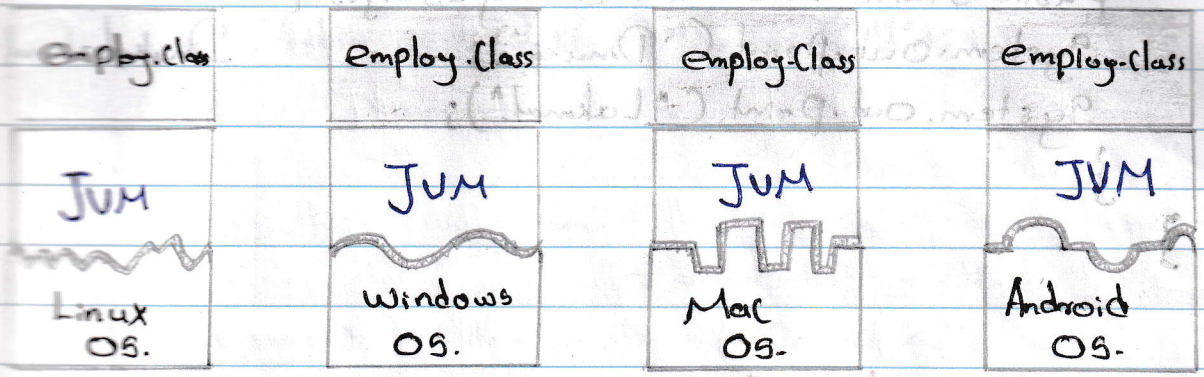
Java supports multi threading. It is a Java feature that allows Concurrent execution of two or more parts of a program or maximum utilization of the cpu.

⊛ Simple. →.

Java is a one of the simple Language s as it dose not have Complex features like pointers, Operator, overloding, multiple inheritanceg, and Explicit memory allocation.

No:_____ Date: __/__/____

Why is the JVM not platform independent?



The key here is that the Jvm depends on the Operating system - So if you are running Mac OS X you will have a different JVM than if you are running windows or some other opparating system. This fact can be verified by trying to download the Jvm for your machine - when ~~you~~ trying to download it, you will be given a list of JVM's corresponding to different operating systems, and you will obviously pick whichever JVM is targeted for the Operating System that you are running

Print vs Println

- Print().→
  Print is given text or object without adding a new line.
- Println() →
  Print ~~object~~ the given text object and then moves to the next line.

example.

Java Code.	Output.

Public Class Example {
Public Static void main (String args []){
System.out.println ("Dinuka");
System.out.println ("Lakmal");
	Dinuka
	Lakmal.

No : _____ Date: __/__/____

```
Public Class Example {
    Public static void main (String [] args) {
        System.out.print ("Dinuka");
        System.out.print ("Lakmal");
    }
}
```

Dinukalakmal

Main method declarations

* Valid main method declarations.

```
Public Class Example {
    Public static void main (String args[]) {
        System.Out.Println ("Hello world");
    }
}
```

* In this Java Code this is a valid and there is no any error for this programing Code.

```
Public Class Example {
    Public static void main (String [] args) {
        System.out.println ("Hello world");
    }
}
```

* In this Java code this is a valid there have change the "(String [] args)" in there there is no any error for this Programming Code.

No :_____ Date: __/__/_____

```
Public Class Example {
    Public Static void main (String [] gunapala){
        System.out.println ("Hello world");
    }
}
```

* In this Java program this is a valid and there have Change the "(String [] gunapala)" in there is ~~there~~ no any error for this programming code.

```
Public Class Example {
    Static Public void main (String [] args) {
        System.out.println ("Hello world");
    }
}
```

* In this Java program this is a valid and there have Change the "Static public void main" and in there is no any errors for this Java code.

* Invalid main declarations.

```
Public Class Example {
    Static void main (String [] args) {
        System.out.println ("Hello world");
    }
}
```

Public ?

// error.

* Invalid main methord in this Code there we Cant See any error when we Compile by Javac but when we open it their have a error massage as like "Main method not found in Class Example."

No: _____ Date: ___/___/_____

⊙ Public Class Example {
    Public void main (String []args){
        System.out.println("Hello world");
    }
}

⊙ Public class Example {
    Public Static void main (String args) {
        System.Out.println ("Hello world");
    }
} //error

⊙ Public class Example {
    Public Static void Main (String []args){
        System.Out.println ("Hello world");
    }
}

* Illegal main method declarations

⊙ Public Class Example {
    Public Void Static main (String []args) {
        System.Out.Println ("Hello world");
    }
} //compile error.

* In this Java code when we compile it
There have been Compile errors

⊙ Public Class Example {
    Public Static main (String []args){
        System.Out.println ("Hello world");
    }

No:

Date: __/__/____

Java Comments

* Inline Comments - we use "//" for the inline comments.

```
Public class Example {
    Public static void main (String [] args){
        //methana A akura Print kara eka wenne
        System. out. println ("A");
    }
}
```

* Inthis Java program in this line 3. is a Comment. Its not give output ~~when we~~ and when we Compile it the compiler skip that line.

* Block Comments - we use "/* */" for the block Comments.

```
Public class Example {
    Public static void main (String [] args) {
        System.Out.println ("A");
        /* This lines are
           not Compiled by
           JavaC */
        System.Out.println ("B");
    }
}
```

These lines are not compile by JavaC. Because of the Block coments

No: _____ Date: __/__/____

⊛ Java Literals.

Any constant value which can be assigned to the variable is called literal / constant.

what are the literals in java,.

* Integral literals.
* floating point literals.
* Character literals.
* Boolean literals.
* String literals.

⊙ Integer Literals. →

we can write numbers without a fractional or decimal Component. there have a four types for write the Integer. (3, 2, 1, 0, -1, -2, -3)

⊙ decimal Integer literals. → In this form allow digits are 0-9.

ex :-

```
int x = 10;
```

⊙ Binary Integer literals → in this format we need to add "0B" for the before the add values.

ex :-

```
System.out.println(0B1010);
```

(0B: Binary value.)

Output → 10 (decimal value printed.)

⊙ Octa Integer literals → in this format we need to add "0" ~~for the be~~ before we add the values.

No :

Date: __/__/____

⊙ Hexa-decimal Integer literal. ⟶ In this format we need to add. "0x" for befor the Hexadecimal.

ex: System.out.println (0xA);

Output ⟶ 10

⊙ Floating-points Literals.

floating points numbers are used to represent numbers that have a desimal point in them (such as. 9.6 or 96.9999)

```
Public class Example {
    Public static void main (String [] args) {
        System.out.println (10.5);
        System.Out.println (-10.5);
    }
}
```

Output ⟶ 10.5
-10.5.

⊙ Characters.

when we use the Characters we need to use (' ') marks for the Java program. In this Characters. we Can write One letter for the Characters.

```
Public Class Example {
    Public static void main (String [] args) {
        System.out.Println('A');
        System.Out.Println('@');
        System.Out.Println(' ');
    // System.out.println ('AA');  // Illegal.
    }
```

No: _____ Date: __/__/____

⑧ Special Characters.

/n → new line Character.
/t → tab Character.
/ → back Slash Character.
/b. → back Space.

we Cant print these Special Characters as like using normal method.

/n → new line Character.

```
Public Class Example. {
    Public Static void main (String args []){
        System.out.println ("my name is /n -
        - Dinuka");
    }
}
```

Output → - My name is
- Dinuka.

/T → tab Character

```
Public Class Example {
    Public Static void main (String [] args){
        System.out.println (" Hello /T Java ");
    }
}
```

Output → - Hello Java

No : _____ Date: __/__/____

/ → backslash Character.

```
Public Class Example {
    Public static void main ( String [] args ) {
        System.out.println ('\"");      // "
        System.out.println ("\\");      // \
        System.out.println ("\"JVM\"");  // "JVM"
    }
}.
```

Output. → "
\
"JVM"

⊙ Boolean literals.

Boolean can store true or false values.

```
Public Class Example. {
    Public static void main (String [] args) {
        System.Out.println ( 10>5);
        System.out.println ( 3>10);
    }
}
```

Output - True.
False.

⊛ Computer variables

In Java The variables are mean Containers for data values temporary.

INCOMPLETED
Date: 21/03/2024
Signature:

A variable is use for store data temporary and It's store data on our volatile memory (RAM).

No: _____ Date: __/__/____

* Declaring (Creating) variables.

To create a variable you must speify the type and

ex :-

```
int x;
```

int → Variable type.
x → Variable.

* Variable initialization.

after the we declare the variable we can add values. fo that variable.

ex :-

```
x = 10;
```

x → Variable
10 → value.

* Variable declaring and initialization.

also we can do this declaring and initialization in one time.

ex :-

```
int x = 10;
```

int → Variable type.
x → variable.
10 → value.

examples for variables.

```
Public Class Example {
    Public Static void main (String args []) {
        System.Out.Print ("Start program");
        // Computer variables.
        int x = 80;
        int y = 70;
```

No: Date: __/__/____

```
        System.out.println(x);
        System.out.println(y);

        System.out.println.(x+y);
        System.out.println(x-y);

        System.out.println(x+to);

        System.out.println("end program");

    }
}
```

Output →
Start program
80
70
150
10
90
end program

Case No	Examples	Output.

Case I	`Public class Example {`
	`  Public static void main (String args [ ]) {`
	`  int x ; //variable declaration.`
	`  x = 20; // variable Inetialization`
	`  Int y = 10;`
	`  }`
	`}`
		20

Case II	`Public class Example. {`
	`  Public static void main (String args [ ]) {`
	`  int x ;`
	`  x = 20;`
	`  System.out.println (x);`
	`  }`
	`}`
		20.

No :_____ Date: __/__/____

Case III

```
Public Class Example.{
    Public Static void main (String args [])){
        int x;
        x = 10;
    System.out.println (x); //10
    System.out.println (x);
    }
}
```

10
10

Case IV

```
Public Class Example {
    Public Static void main (String args [])){
        int a;
        System.out.println (a);
    }
}
    //Variable a might not have been initialized.
```

Error

Case V

```
Public Class Example {
    Public Static void main (String args [])){
        int a;
        System.out.println(a);
        a = 10;
    }
}
```

Error

when printing The Java program is run by, 1st to 2 line by line.

Case VI

```
Public class Example {
    Public Static void main (String args [])){
        int a;
        a = 100;
        a = 200;
```

200.

No : _____ Date: __/__/____

Case VII

```
Public Class Example {
    Public Static void main (String args[]){
        int a;
        a = 100;
        System.out.println(a);
        a= 200
        System.out.println(a);
    }
}
```

100
200

Case VIII

```
Public Class Example {
    Public Static void main (String args[]){
        int a;
        a = 100;
        int b = 200;
        b = a // 200 = 100
        System.out.println(a);
        System.out.println(a);
    }
}
```

100
100

Case IX

```
Public Class Example {
    Public Static void main (String args[]);
        int a,b,C ;
        a = 10;
        b = 20;
        C = 30;
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
    }
}
```

10
20
30

No : _____ Date: __/__/____

Case X

```
Public class Example {
    Public Static Void main (String args [] ) {
        int a,b,c;
        a = b = c = 100;
        System.out.println (a);
        System.out.println (b);
        System.out.println (c);
    }
}
```

100
100
100

Case XI

```
Public Class Example {
    Public Static void main (String args [] ) {
        int a = 100, b = 200, c = 300;
        System.out.println (a);
        System.out.println (b);
        System.out.println (c);
    }
}
```

100
200
300

Case XII

```
Public Class Example {
    Public Static void main (String args [] ) {
        int a = 100, b,c = 300;
        b = 20;
        System.out.println (a);
        System.out.println (b);
        System.out.println (c);
    }
}
```

100
20
300

Case XIII

```
Public Class Example {
    Public Static void main (String args [] ) {
        int a;
```

Error

```
        a = 100;
        a = 200;
        System.out.println(a);
    }
}
```

```
Public Class Example. {
    Public Static void main (String args [])){
        int a= 10;
        int b= 20;
        int c = a+b;
        System.out.println (c);
    }
}
```

Output: 30

1) Code Block.

A Code block is a Selection of Code enclosed in Curly braces ({}). Code block are used to goroup one or more Statements. Code blocks Can be nested.

example :-

```
{
    int x = 10;
}
```

example.

```
Public Class Example {
    Public Static void main(String args []){
        int x = 10;
        System.out.println (x);
        { // this is a code block {...}.
            int y = 20;
```

Output:

10
20

No : _____ Date: __/__/____

⑧ Variable Scope & Lifetime.

The Scope of a variable refers to the areas or the section of the program in ~~which~~ which the variable can be accessed, and the life time of a variable indicates how long the variable stays alive in the memory.

Example.

Output

```
Public Class Example {
    Public Static Void main (String args []){
        int x = 10;
        System.out.println(x);
        {
            int y = 20;
            System.out.println (y);
            System.out.println(x); // This is in main block and their have'nt any Errors
        }
    }
}
```

Output:
10
20
10

```
Public Class Example {
    Public Static void main (String args[]){
        int x = 10;
        System.out.println(x); // line 1
        { int y = 20;
            System.out.println (y); //line 2
            System.out.println (x); //line 3
        }
    System.out.println (x); //line 4
    System.out.println (y); //line 5
    }
}
```

Error

(In the line 5 that varriable cannot be find

No: _____ Date: __/__/____

Final keyword.

final variables are ~~Const~~ nothing but Constants we Cannot Change the value of final Variable once it is initialized.

Example :

```
Public class Example {
    Public Static void main [String args []) {
        final int days = 7;
        System.out.println.(days);
    }
}.
```

Output.

7

```
Public class Example {
    Public Static void main (String args []){
        final int days = 7;
        System.out.println (days);

        days = 8; // Cannot be asign a value to final variable.
        System.out.println (days);
    }
}. (There have an Error because we can't add values for final variables. after it initialized.).
```

Error.

No : _____ Date: __/__/____

(*) String Concatenation vs Arithmetic addition.

"+" operator in Java can be used to add numbers and Concatenate strings. following rules should be Considered.

* Only numbers as operands then result will be a number.

  (8 + 2) = 10 //.

* Only String as operands then result will be a Concatenated String.

  "Nimal" + "kamal" = "Nimalkamal" //

* If both numbers are String ~~and operands~~ number operands., then numbers comming before String will be treated as numbers.

  (1 + 2 + "Dinuka" + "Lakmal") = 3Dinuka Lakmal

* If both numbers and Strings as operands then numbers coming after String will be treated as ~~numbers~~ String.

  (1 + 2 + "Dinuka" + "Lakmal" + 4 + 5) = 3Dinuka Lakmal 45

* Above rule can be Overridden using brackets ().

  (1 + 2 + "Dinuka" + "Lakmal" + (4 + 5)) = 3Dinuka Lakmal 9

No: _____ Date: __/__/____

Example. →

```
Public Class Example {
    Public Static void main. (String args [ ]) {
        System.out.println (10 + 20 );
            // Aritmatic Addition ⇒ 10 + 20 = 30.
        System.out.println ("Dinuka" + "Lakmal");
            // String Concatination. "Dinuka" + "Lakmal" = "DinukaLakmal"
    }
}.
```

④ Keyboard input.

Scanner. →.

In Java, "Scanner" is a Class in the "Java.util" Package that allows you to read input of various types from different input sources, Such as the keyboard files, Or Strings. It provides methosdes to parse primitive types and strings from inpat streams.

```
import java.util.*;
Public class Example {
Public Static void main (String args [ ]) {
        Scanner input = new Scanner (System.in);
        int Count = input.nextInt ( );
        System.out.println ("Today accident Count :" + Count);
    }
}
```

Output. →

We need to insert Count. we can see that the Courser

No: _____ Date: __/__/____

Example

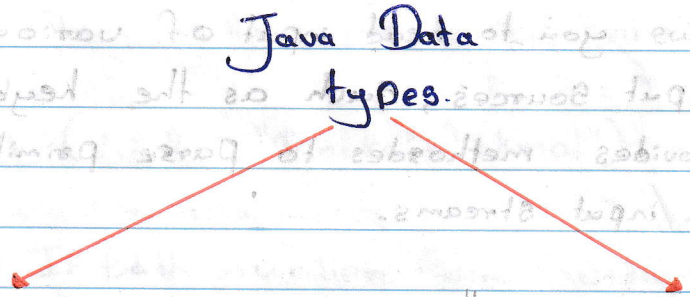
```
import java.util.*;
Public Class Example {
    Public static void main (String args []){
        Scanner input = new Scaner (System.in);
        System.out.println ("Input daily count:");
        int Count = input.nextInt ();
        System.out.println ("Today daily Count:"+(count);
    }
}
```

Output

```
Input daily count: 10
Today accident count:
```

2024/03/21

Java data types.



Primitive data type

* A primitive data type Specifies the size and type of variable values and it has no additional methods.

- byte
- Short
- int
- long
- float
- double
- boolean

NOn - Premitive data type.

* Non primitive data types are Called reference types because they reffer to object.

- String
- Arrays
- Classes.

No : _____ Date: __/__/____

① Primitive Data types

The types of what we can store data tempary in variable.

→ ② Integer. -

An integer is a whole number that can be positive negative or zero.

→ byte.
- * Size → 1byte
- * Stores whole numbers from -128 to 127.

→ Short
- * Size → 2 bytes
- * Stores whole numbers from - 32,768 to 32,767.

→ int.
- * Size → 4bytes
- * Stores whole numbers from -2,147,439,648 to 2,147,438,647.

→ Long
- * Size → 8 bytes
- * Stores whole numbers from -9,233,372,036,854,775,808 to 9,233,372,036,854,775,807

MIN		MAX
-2^7	byte	2^7
-2^{15}	Short	2^{15}
-2^{31}	int	2^{31}
-2^{63}	long	2^{63}

No : _____ Date: __/__/____

→ * Floating Point. —

floting point types represents numbers with a factional part, containing one or more decimals. There are two types
float and double

→ float

* Size - 4 byets.
* Stores fractional numbers Sufficent for Storing 6 to 7 decimal digits.

→ double.

* Size - 8 bytes.
* Stores fractional numbers. Sufficent for Storing 15 decimal digits

→ * Boolean.

Very often in programming we need datatype that can only have one vales as like
YES, NO / ON, OFF / TRUE, FALSE
for this boolean datatype which can only take true or false.

→ boolean

* Size - 1 bit
* Stores true or false value.

→ * Char

The Char datatype is cesed to store a Single Character The Character must be Surrounded by Single quotes, like 'A', 'B'...

No: _____ Date: __/__/____

→ Char.

* Size - 2bytes.
* Stores a single Character / letter or ASCII values.

① Numerical data types.

byte → 8bit → -2^7 → 2^7-1
Short → 16bit → -2^{15} → $2^{15}-1$
int → 32bit → -2^{31} → $2^{31}-1$
Long → 64bit → -2^{63} → $2^{63}-1$
float → 32bit → IEEE 754 Standard for respecting floating Point numbers.
double → 64bit → IEEE 754 Standard for respecting floating Point numbers.
Char. → 16bit → 0 → $2^{16}-1$

Examples. for data types.

example.

→ Primitive data types

```
Public class Example {
    Public static void main (String args[]){

    byte b1 = 10;
    System.out.println(b1);

    Short s1 = 10;
    System.out.println(s1);

    int i1 = 10;
    System.out.println(i1);
```

Output.

```
10
10
10
10
```

No : _____ Date: ___/___/_____

Floting points Pegmary data types

```
Public Class Example {
    Public Static void main (String args [ ]){
        ~~float~~
        float f1 = 10.5f; //by defalt think as doble value.
        System.out.println ( f1);
/* when add the flote value we need to add the "f" after the value */
        double d1 = 10.5;
        System.out.println (d1);
// floting point default data type is -> double.
    }
}
```

10.5

10.5.

Boolean primitive data types

```
Public Class Example {
    Public Static void main( String args [ ]){

        boolean b1 = true;
        System.out.println (b1);

        boolean b2 = false;
        System.out.println (b2);

        boolean b3 = 10>5 ; //condition
        System.out.println (b3);

    }
}
```

true

false

true.

No: _____ Date: __/__/____

Another Primitive data types.

```
Public Class Example {
Public Static void main (String args [ ]) {

    Char ch1 = "A";
    System.out.println(ch1);      //A

    Char ch2 = 65
    System.out.println(ch2);      //A

}
```

A

A.

```
Public Class Example {
Public Static void main (String args [ ]) {
    Char ch1 = 65+10;      // 75 = K
    System.out.println(ch1);
}
```

K.

Special note.

When we write the big number on our Java Program we can add unerscore "_" for any number. It's use full to improve readability on our Java programm. It's a new feature in JDK

Example.

Need to add this number to our Java program.

1040 5869365 426 425689 (This Number is very large and It's hard to read to us)

we can write this number like this.

1040_5869_3694_2642_5689

No :_____ Date: __/__/____

⊛ Non Primitive data types

Non primitive data types are called referance types because they refer to objects.

~~de~~ diffeance between non primitive and primitive data types.

⊛ Primitive data types are allredy defined in java Non primitive types are created by the programmer and is not defined by java (expect for String).

⊛ Non primitive types can ~~can~~ be use to call methods to perfom certain operations, while primitive types cannot.

⊛ A primitive type has allways a value. but in the non primitive can be null.

⊛ A primitive type ~~can~~ start with a lowercase letter while non primitive types start with an uppercase letter.

Examples for the non primitive datatypes.

- Strings
- Arrays
- Classes.
- Interface.

Example

```
Public Class Example {
Public Static void main (String args [=]){
    //non primitive datatype.
String name = "kamal";
System.out.println (name);
}
}
```

Output.

```
kamal.
```

No: _____ Date: __/__/____

MIN Values and MAX Values.

we Can find the all data types min values and max values by useing Java Code. example. If you think yo need to find Min value and max value in byte tye. you can. try this.

```
System.Out.println(Byte.MIN_VALUE);   //-128
System.Out.println(Byte.MAX_VALUE);   // 127
```

example.

```
Class Example {
public Static Void main(String args[]){
byte b1 = Byte.MIN_VALUE ; // -128
byte b2 = Byte.MAX VALUE; // 127
System.Out.println("byte min value:" + b1);
System.Out.println("byte max value :" + b2);
```

Output.

```
byte min value :-128
byte max value : 127
```

```
Short S1 = Short.MIN_VALUE; // -32768
Short S2 = Short.MAX_VALUE; // 32767
{ System.Out.println("Short min value :" + S1);
System.Out.println("Short max value:" + S2);
```

```
Short min value: -32768
Short max value:
32767
```

```
int i1 = Integer.MIN_VALUE; // -2147483648
```

Integer min value -2147483648

No :_____ Date: __/__/_____

```
Long l1 = Long.MIN_VALUE; // -9223372036854775808.
Long l2 = Long.MAX_VALUE; // 9223372036854775807
System.out.println("Long min value :" + l1);
System.out.println("Long max value :" + l2);
```

Long min value : -9223372036854775808
Long max value : 9223372036854775807

```
float f1 = Double float.MIN_VALUE; //1.4E-45.
float f2 = float.MAX_VALUE; // 3.4028235E38.
System.out.println("float min value :" + f1);
System.out.println("float max value:" + f2);
```

float min value : 1.4E-45
float max value : 3.4028235E38

```
double d1 = Double.MIN_VALUE; // 4.9E-324
double d2 = Double.MAX_VALUE; //1.79769313486231570E308.
System.out.println("double.min value:" + d1);
System.out.println("Double.max value.:" + d2);
```

double min Value = 4.9E-324
double max value = 1.79769313486231570E308

⊛ Reserved words, keywords and Identifiers.

⊛ Reserved words and ~~key~~ key words.

The Java has set of keywords that are reserved words that Canot be use as the variables, methods, Classes or any other identifiers.

In Java Programming their have 51 ~~Reserved~~ words and ~~At~~ in their the 48 words are the key words.

false, true, null } not keywords

Some key words in Java :

while	public	instanceof	final
do	privet	protalt	Catch
default	Switic	void	Package

No: _____ Date: __/__/____

① Identifiers.

we Can't Change the keywords . but we Can Change identifiers.

- Variable, Methods, Classes., Interfaces., Enum.

example. →.

```
public Class Example {
public static void main (String args []){
    int Sumana.; //keyword = int | identifier = Sumana.
  }
}
```

01

```
Class Example {
static void main (String args []){
[a-z | A-Z | $ | _]
  int x;
  int A;
  int 0; // Illeagal.
  int #;
  int _; //Illeagal.
  int #a;
  int _x;
} .
```

① we cant add underscore or numbers for Identifiers. frist letter.

02.

```
Class Example {
public Static void main (String args[]){
  int Curunt_age. // Snake Case . Specialy using Mysql
  int Curunt Age. // Camel Case. Specialy use in Java
}
```

No: _____ Date: __/__/____

Rule no 03.

```
Public Class Example {
    Public Static void main (String args [ ]){
        int abcdefghIjklmnoparst;
        abcdefghIjklmnoparst = 10;
        System.out.println (abcdefghIjklmnoparst ); // 10,
    }
}.
```

* We Can use any Length for Identifiers.

Rule no 04

```
Public Class Example {
    Public static void main (String args [ ]){
        int a;
        int A;
    }
}
```

Rule no 05.

```
Public Class Example {
    Public static void main (String args. [ ]){
        // Class name -> frist letter should be capital
        // method -> frist name should be simple. with ( )
        // variable -> frist letter should be simple.
        int age;
    }
}
```

* Not Illeagle but It's a standerd.

Rule no 06.

```
Public Class Example {
    Public static void main (String args [ ]){
        int Class; // Illeagle
        int public; // Illeagle
```

* we Cant use recerved words for

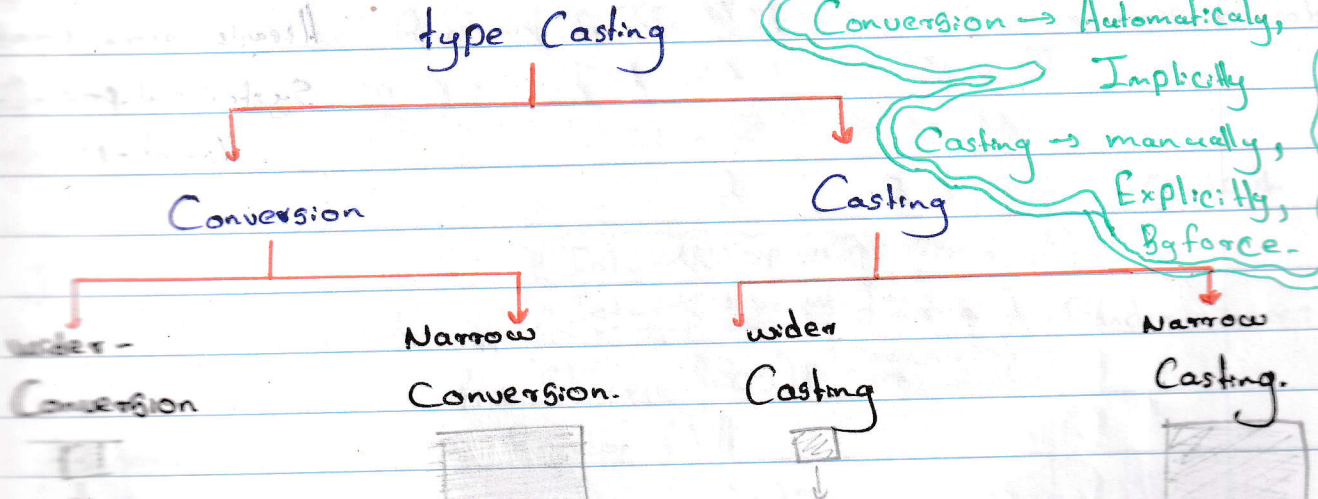
No: _____ Date: __/__/____

1) Keyboard inputs for different data types.

```
import java.util.*
public class Example {
    public static void main(String args[]){
        Scanner input = new Scanner(System.in);

        byte b1 = input.nextByte();
        short s1 = input.nextShort();
        int i1 = input.nextInt();
        long l1 = input.nextLong();

        float f1 = input.nextFloat();
        double d1 = input.nextDouble();

        boolean b1 = input.nextBoolean();

        String s1 = input.nextLine(); // input.next.
    }
}
```

2) Type Casting. → assign one data type to another data type.

- wider → Small to Large
- Narrow → Lage to Small
- Conversion → Automaticaly, Implicitly
- Casting → manually, Explicitly, Byforce.



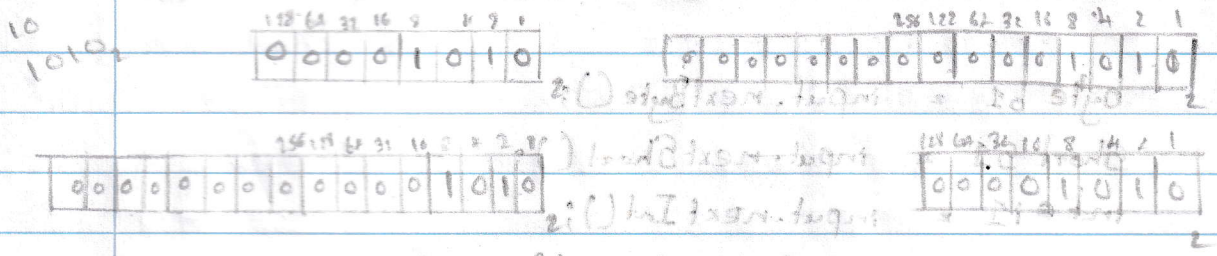
No : _____ Date: __/__/____

* Wider Conversion.

```
byte b1 = 10;
Short S1 = b1;
System.out.println(b1);
```

* Narrow ~~Conversion~~ Casting.

```
Short S1 = 10;
byte b1 = (byte) S1;
System.out.println(S1);
```



* Wider Casting.

```
int total = 115;
int Count = 10;
double result = (double) total/Count;
System.out.println(result);
```

Narrow Convertion

```
byte b = 10;
Short s = 100;
// b = b + s // Illeagal
```

```
b = s // Leagle → Narrow Conversion
System.out.println(b);
// print 110
```

```
b = 10;
S = 245;
```

~~b = s~~ b += s

```
// leagle narrow Conversion.
System.out.println(b);
// print -1;
```

COMPLETED
Date: 28/3/2024
Signature:

No : Date: 28 / 03 / 2024

Spacial note

when we write Java programe using Scanner inputs we can ~~whn~~ write that Java Code two ways.

I methord.

```
import Java.util.*;      // * → All
  Public Class Example.{
      Public Static void main (String args[]).{
      Scanner input = new Scanner(System.in);
    }
  }.
```

// At in the 'I' methord. there happen Java > util all files are open and that all are redy to come wen we call them.

II methord.

```
import Java.util.Scanner;
  public. Class Example {
     Public static void main.(String args[]){
     Scanner input = new Scanner (System.in);

     }
  }.
```

// In second methord their happen the Java > util > Scanner fill is reddy to come when we call. not the all files at there.

⊛ Random Inputs

When we use Random inputs for the auto ganarate numbers

examples.-

Java Code.	Output

import Java.util.*;
 Public Class Example {
 Public static void main (String args[]){
 Random rand = new Random.();
	Random number : 876364

Int_MAX VALUE
Int_MIN VALUE
ගත හැකි උපරිම/අවම අගය

```
import java.util.*;
Public Class Example{
   Public Static void main (Stringargs[]){
      Random rand = new Random();

      int jumpCount = rand.nextInt(50);
      System.out.println("Jump count is " + jumpCount);
   }
}
// In this java code we have given the range
0 — 50, the java random is generate
0 — 50 between number for us.
```

range. we have add.

Jump Count is 40

Java Operators

Operators are Symbols that perform Operations on variables and values.

Operator precedence

Operators	Precedence.
postfix	expr ++, expr --
unary	++ expr, -- expr, +expr -expr ~ !
multiplicative	* / %
additive	+ -
Shift	<< >> >>>
relational	< > <= >= instanceof
equality	== !=
bitwise AND	&
bitwise exclusive or	^
bitwise inclusive or	|
Logical and	&&
Logical or	||

No : _____ Date: __/__/____

Operators in Java can be Classified into 5 types.

* Arithmatic Operators.
* Assignment Operators.
* Relational Operators.
* Logical Operators.
* Unary operators.
* Bitwise. Operators.

① Arithmatic Operators. –

Arithmatic operators are used to preform arithmatic operations on variables and data.

Operator.	Operation.
+	Addition. (එකතුකිරීම)
-	Subtraction (අඩුකිරීම)
*	Multiplication. (ගුණකිරීම)
/	Division (බෙදීම)
%	Modulo Operation. (ඉතිරිය)

Examples.

Examples.	Output.

```
Public Class Example {
    Public Static void main (String args [ ]){
        int x;
        x = 10 + 20;        // Arithmatic addition.

        name = "Saman" + "Perera" ;   // String Concatnation
    }
}.
```

Additive, Addition.

```
Public Class Example {
```

No : _____ Date: __/__/____

```
        int x;
        x = 20 - 10;
        System.out.println (x);
    }
}
```

Subtraction

10

```
Public Class Example {
    Public Static void main (String args[]){
        int x;
        x = 20 * 10;
        System.out.println (x);
    }
}
```

multiplication.

200

```
Public Class Example {
    Public Static void main (String args[]){
        int x;
        x = 20 / 10;
        System.out.println (x);
    }
}
```

Division

2

$$10 \overline{)\,20\,} \quad 2,\ 20,\ 0$$

```
Public Class Example {
    Public static void main (String args[]) {
        int x;
        x = 25 % 10;
        System.out.println (x);
    }
}
```

Modulo operation

5

$$10 \overline{)\,25\,} \quad 2,\ 20,\ 5$$

No : _____ Date: __/__/____

⑧ Assignment Operators

Assignment operators are used in Java to assign values to variables

Operators	Example	Equivalent to
=	a = b;	a = b;
+=	a += b;	a = a+b;
-=	a -= b;	a = a-b;
*=	a *= b;	a = a*b;
/=	a /= b;	a = a/b;
%=	a %= b;	a = a%b;

Example.

```
Public Class Example {
Public Static void main(String args[]){

    int x;
    x = 20;
    x += 10;    // x = x+10

    System.out.println(x);    // 30

    x -= 10;    // x = x-10
    System.out.println(x);    // 20

    x *= 2;     // x = x*2
    System.out.println(x);    // 40.

    x /= 2;     // x = x/2
    System.out.println(x);    // 20
```

Compound operators.

Output

30
20
40
20
5.

No : _____ Date: ___/___/_____

⊛ Java Relational operators

Relational operators are used to check the relationship between two operands.

Operator	Discription.	Example.
==	Is Equal to	9==5 → false
!=	Not Equal to	3!=5 → true.
>	Greater than	3>5 → false
<	Less than	3<5 → true
>=	Greater than or Equal to	3>=5 → false
<=	Less than or Equal to.	3<=5 → true

Example

```
Public class Example {
    public static void main (String [] args){
        int a = 10;
        int b = 20;
        System.out.println ( a == b); // equal
    }
}.
```

10 == 20 (සමාන නොවේ.)

Output.

false.

```
Public class Example {
    Public Static void main (String [] args){
        int a = 20;
        int b = 20;
        System.out.println ( a != b); //not equal
    }
}.
```

20 != 20 (සමාන නොවේ).

False.

```
Public class Example {
    Public Static void main (String [] args) {
```

No : _____ Date: __/__/____

```
        b = 20;                                    True.
        System.out.println( a>=b); // greter than and equal.
    }
}.                                  20 >= 20
```

~~Public Static voi~~

```
Public Class Example {
  Public Static void main (String [] args) {
    int a = 10;                                    // false.
    int b = 20;
    System.out.println (a>b); // greter than!
  }                          10 > 20   // false.
}.
```

```
Public Class Example. {
  Public Static void main (String [] args) {          false.
      int a = 20;
      int b = 10.;
      System.out.println ( a<=b); // less than and equal.
  }                  20 <= 10
}.
```

```
Public Class Example {
  Public Static void main (String [] args){                True.
      int a = 10;
      int b = 20;
      System.out.println ( a<b); // Less than.
  }                      10 < 20
}.
```

No: _____ Date: ___/___/_____

⊛ Java Logical Operators.

Logical operators are used to check whether an expression is true or false

Operator	Example	Meaning
&& (Logical And). ShortCircuit and	expression1 && expression2.	true only if both expression1 and expression are true.
|| (Logical Or). Shortcircuit or	expression1 || expression 2	true if either expression 1 or expression 2 is true.
! (Logical. NOT)	!expression	true if expression is false and vice versa

Examples.

```
Public Class Example {
    public static void main (String [] args) {
        int x = 10;
        int y = 20;  // Short Circuting and.

        boolean b1 = x > y && x == y; // false && false
        System.out.println (b1);
    }
}.
```

Output:

False.

↳ false.

```
Public Class Example {
    Public Static void main (String [] args) {
        int x = 10;
        int y = 20          // Operator or. and
        boolean b1 = x < y & x == y;
                    // 10 < 20 & 10 == 20.
                    True & False.
```

False

No :

Date: __/__/____

```
Public Class Example {
Public Static void main (String [] args.){
    int x = 10;
    int y = 20;                    // Short Circuting OR.
    System.out.println (
    boolean b1 = x<y || x==y ;
                  // 10<20 || 10==20
                  // True || false.   // True.
    System out.println (b1);
}
}
```

True.

By useing Short Circute Operators It is usefull for the Increase your Program Proformence.

```
Class Example {
Public Static void main (String [] args){
    int x = 10;                    // Operator Or.
    int y = 20;
    boolean b1 = x<y | x==y;
    System.out.println( b1);       // 10<20 | 10==20
}                                  // TRUE | False.
                                   // TRUE
```

TRUE.

* AND.

B.	A+B.(AORB)	A	B.	A.B (AandB).
0	0	0	0	0
1	1	0	1	0
0	1	1	0	0
1	1	1	1	1

* XNOR.

B	A⊕B.	A	B	$\overline{A \oplus B}$
		0	0	1

No :_____ Date: __/__/____

⊛ Java unary operators.

Unary operators are used with only one operand

Different types of unary operators are...

Operator.	Meaning
+	Unary Plus not necessary to use since numbers are positive without using it.
−	Unary minus. inverts the sign of an expression.
++	Unary Operator. Incriment. Incriments value by 1.
−−	Decrement operator. decrements value by 1
!	Logical Complement operator. inverts the value of a boolean

Example.

Output.

⊛ Unary plus operator

```
Public Class Example {
    Public Static void main (String [] args){
        int x = +10;
        System.out.println (" x:" + x);
    }
}.
```

x:10

⊛ Unary minus.

```
Public Class Example {
    Public Static void main (String [] args){
        int x = - 10;
        System.out.println (" x:" x);
```

x:-10.

No: _____ Date: __/__/____

x++	x--
x = x+1	x = x-1
++x	--x
1+x = x	1-x = x.

① unary Incriment Operators

```
public Class Example {
    public Static void main (String [] args){
        int x = 10;                                   x: 11
        x++; // 10+1 = 11                            x: 12.
        System.out.println("x:" + x);
        x++; // 11+1 = 12
        System.out.println ("x:" + x);
    }
}
```

② unary decrement Operators.

```
Public Class Example {
    Public Static void main (Static [] args){          x: 9
        int x = 10;                                    x: 8.
        x--; // 10-1 = 9
        System.out.println ("x:" + x);
        x--; // 9-1 = 8.
        System.out.println ("x:" + x);
    }
}
```

③ Logical Compliment Operator.

```
Public Class Example {
    public Static void main(String [] args){          True.
/* ! → Logical compliment Operator → inverts the      False.
   value of a boolean. */
        boolean a = false;
        System.out.println (!a);
        // false → Invert. == True
        boolean b = True;
        System.out.println (!b);
    }                  // True → Invert == False.
```

INCOMPLETED
Date: 10/4/2024
Signature:

No : _____ Date: ___/___/_____

$--x$ $1-x = x$	$++x$ $1+x = x$
$x--$	$x++$ $x = x+1$

⑧ Java bitwise operators.

Bitwise operators in Java are used to preform Operations on individual bits

Operator.	meaning.
~	Bitwise Complement.
<<	Left Shift.
>>	Right Shift.
>>>	UNsigned Right Shift.
&	Bitwise and
|	Bitwise inclosive OR.
^	Bitwise exclusive OR.

But when we programming we also using shortcut and and or (&&, ||) Because it is increase our Java programs

⑨ Java flow Controllers

* If Statement → Java if statement is the most simple decision - making statement. we use Java statements for the Certain Conditing. are true then a block of statment is executed and otherwise not.

Condition

```
if (x > 5) {
    System.out.println ("x");
} else {
    System.out.println ("0");
}
```

If the Condition is true → Body of the Statements.

If the Conditon is false → Body of the Statements.

No : _____ Date: __/__/____

Examples.

Output

```
Public Class Example {
    Public Static void main (String [] args) {
        int x = 3;
        if ( x > 5) {
        System.out.println ("A");
        }
        System.out.println ("B");
    }
}
```

B.

If Condition

```
Public Class Example {
    Public static void main (String[] args) {
        int x = 3;
        if (x > 5) {
        System.out.println ("A"); // If the Condition true
        } else {
        System.out.println ("B"); // If the condition false.
        }
    } /* At in the else condition we Cant ad new
}. Conditing for it we can only use the if Condition's for
   false --- */
```

B.

else Condition

```
Public Class Example {
    Public Static void main (String [] aggs) {
        int x = 3;
        if (x > 5) {
        System.out.println ("A"); // If true the fors condition print as "A"
        } else if ( x > 0) {
        System.out.println ("B"); // If true the second condition print as the "B"
        } else { System.out.println ("lower than zero");
        }.    } // If didint true take any statement
              // Print as the ("lower than zero")--
    }
```

B.

else if Condition

No : ______ Date: __/__/____

(*) Ternary Operator

There is also a short hand if, else, which is known as ternary operators. because it consist of tree operands. It can be use to replace multiple lines of codes with single line. and is the most often used to replace simple if else statement.

```
int age = 18;
~~int ve~~ String resalt; //varable declare.
reslt = (age > 15) ? System.out.println("you are elder person"); : Soutprintln("younger person");
```

reslt ← There have to initialize the variable

(age > 15) → Condition

? → if

: → else

Example. Output.

```
import Java.util.*;
public ~~St.~~ Class Example {
    public static void main (String [] args) {
    Scanner input = new Scanner (System.in);
        System.out.print("Input your age:");
        int age = input.nextInt ();

        System.out.println((age >18) ? "Eligible" : "Not-Elegible");

    } /*If we insert Greter than 18 for the age
        there give out put as the Elegible
        and when we insert the less than number for
        the age there point Not Elegible */
}.
```

*Methord ①

```
import. Java.util.*;
Pablic class Example {
    Public Static void main (String [] args){
```

```
        System.out.print("Input your age");
        int age = input.nextInt();

    String resalt;
      resalt = (age > 18) ? "Eligible." : "Not Elegible.";

      System.out.println(resalt.); // I have make the variable and
    }                              //assign the Jerny operator
}                                  // what I need to do and print variable.

import java.util.*;
public class Example {
  Public Static void main(Strin[]args){
  Scanner input = new Scanner (System.in);

    System.out.print("Input PRF Marks :");
    double prf = input.nextDouble();
    // Declare the variable in string datatype because need to print string data.
    String grade = (prf >= 75) ? "A" :(Prf >= 65) ? "B" :(Prf >= 55)?
                 -> "C" : "fail"

      System.out.println(grade);

  }
```

No : _____ Date: 04/04/2024

⑧ Java Switch statement

The Switch Statement in Java is a multi-way branch Statement. in Simple words the Java Switch Statement executes one Statement from multiple Conditions.

```
Switch(A){
    Case 1 : System.out.println("1");
}.
```

(සමාන කම)

Example.

```
import java.util.*;
Public Class Example {
    Public Static void main (String[] args) {
    Scanner input = new Scanner (System.in);
        System.out.print("Input Tyer count:");
        int tyerCount = input.nextInt();
        Swith (tyerCount) {
            Case 1 : System.out.println("weelbarrow");
            Case 2 : System.out.println("bicycle");
            Case 3 : System.out.println("Threeweel");
            Case 4 : System.out.println("car");
            default : System.Out.println("Not a vehical.");
        }
    }
}.
```

Output.

we input "1"
weelbarrow

when we input "2"
bicycle.

when we input "3"
Treeweel

when we Input "4"
Car

when input (any other num) without 1, 2, 3, or 4
Not a vehical.

Treeweel
Car
Not a vehical.

Car
Not a vehical

/* At in their. when we input the number there point all of we have given Souts. So we need to use. **break;** keyword at end of the every lines ~~in Swit~~ where at in Switch Statement Cases -- */

No : _____ Date: ___/___/_____

```
import java.util.*;
public Class Example {
Public Static void main (String[] args) {
 Scanner input = new Scanner (System.in);
    System.out.println ("Input ~~your~~ Tyer count:");
    int tyerCount = input.nextInt();
       Switch (tyerCount) {
          Case 1 : System.out.println("weelbarrow");
          ~~Case~~ break;
          Case 2 : System.out.println ("bicycle");
          Case 3 : System.out.println ("treeweel");
          break;
          ~~ca~~default : System.out.println ("Not a vehical");
          break;
       }
}
}
```

when we input "1"
weelbarrow.

when we input "2"
bicycle.

treeweel.

when we input "3"
treeweel.

when we input "1,2,3" else any other number
Not a vehical

we Can add multiple values for one case.

```
------
--------
String Day = input.next();
Switch (Day) {
   Case "Sunday", "Sataday" : System.out.println("freeday");
   break;
   ~~Case~~ defalt : System.out.println("Not a free. day");
--------
```

NEXT PAGE

No: _____ Date: __/__/____

⑧ Arrow Notation, (->)

we don't need add break; Command for the Switch feather...

Example.

```
import java.util.*;
Public Class Example {
   Public Static void main (String [] args) {
    Scanner input = new Scanner (System.in);
      System.out.println("Input a number");
       int num = ~~ment~~.Input.nextInt();
         Switch (num){
           Case 1 -> System.out.println(1);
           Case 2 -> System.out.println (2);
           default -> System.out.println (3);
         }
   }
}
```

Output.

When we input "1"

1

When we input "2"

2

when we Input 1,2 false number

3

```
import Java.util.*;
Public Class Example {
   Public Static void main (String [] args){
    Scanner input = new Scanner (System.in);
      System.out.println("Input number");
      int num = input.nextInt();
        Switch (num) {
            Case 1 -> {
                      System.out.println("Number 1");
                      System.out.println("A");
                      }
            default. -> { System.out.println("Invalid Number");}
        }
```

when we input 1

Number 1

A

when we input any other number.

Invalid Number

No: _____ Date: __/__/____

⑧ Java switch Expression.

Switch Statement vs Switch expression.

Switch Statement with Arrow notation.

```
import java.util.*;
Public class Example {
Public static void main (String[] args) {
Scanner input = new Scanner (System.in);
   System.out.print("Input a number");
   int num = input.nextInt();
    Char ch1;
   Switch (num) {
      Case 1 -> ch1 = 'A';
      Case 2 -> ch1 = 'B';
      Case 3 -> ch1 = 'C';
      default. -> ch1 = 'O';
       }
System.out.println("The character is" + ch1);
```

At in this Java program we have Take input of the numbers. and when we input the numbers their Point Character value.

Switch Expressions with (:) notation.

```
import java.util.*;
Public class Example {
  Public static void main (String[] args){
   Scanner input = new Scanner (System.in);
     System.out.print("Input a number");
     int num = input.nextInt();
     Char ch1 = Switch (num) {
                Case 1 : yield 'A';
                Case 2 : yield 'B';
                Case 3 : yield 'C';
                default : yield 'O';
                 };
    System.out.println("The character is" + ch1);
  }
}.
```

Yield → keyword in Java.
* yield is only used within Switch expressions and it is not a genaral purpose in java.

Special note. → Switch expression

when we input the Switch statement (->) by using arrow notation is this have one State ment it's ok and no need add yield but at in thair have two or many Statments we must add yield keyword and also when we using (:) colon mark we must use yield keyword

At in this Java program. we have take input values and what we input. acoding to the input there Output the Character value.

We can use yield for the variable. becouse it is not the genaral keyword but we Cant use any keyword for

No: _____ Date: ___/___/_____

example-

```
A = Switch.(__) {
    Case "1" -> ..."1"... ; // Ok
    Case "2","3" -> "2"; // error
```

Because when ~~we~~ Their have two or more statements at in their we need to add ~~yield~~ yield keyword.

```
    Case * : "3" // error.
```

when we using colon (:) we must add ~~yield~~ yield keyword

⑧ Java Methords ->

The method in Java or methords of Java Collection of statements that preform some spesific tasks and return the results to the Caller. A Java method Can Proform some specific tasks without returning the ~~code~~ anything Java methords allows us to reuse the Code withe out retyping the code

How Can we found methord.

In method its first letter is Simple and the last there have two brackets ()

example. ↓

* Public static void main (String[] args)
* ~~Scanner input = new Scanner (System.in);~~
* Public static void Count ()

No:

Date: __/__/____

Examples.

Outputs.

```
Class Example {
Public Static void printTotal(){
    double prf = 75;
    double dbm = 78;
    double total = prf + dbm;
    System.out.println("Total :" + total); }.
Public Static void main(String [] args) {
    printTotal( ); // Call printtotal.
}
```

Total : 153.0

Statements. — Method body.

Main method. — Method body

Call to PrintTotal. method

when we Create new method we need to call to that method The Jvm (Java vertual Maghin) is frist ly come to the Java main method Jvm is allways read-fristly. In the Java program Can't have One more Main methods and when we Create the methods after we Create method we need to Call to the method. when we didn't Call to the method there didnt read by Jvm and didn't give outputs for us.

```
Class Example {
Public Static void main (String [] args){
    printHello ();
    printName.();

}
Public Static void printHello ( ) {
    System.out.println ("Hello world");
  }.
public Static void printName (){
    String name = "Dinuka";
    System.out.println(name);
```

Hello world
Dinuka

No :_____ Date: __/__/____

```
import java.util.*;
Public Class Example {
Public static void checkOdd0r Even (){
    Scanner input = new Scanner (System.in);
     System.out.print("Input Nunumber ");
     int num = input.nextInt();

      if (num %2 ==0) {
          System.out.println("num + "Even number");
       } else {
            System.out.println (num + "Odd number");
            }
}
Public static void main (String[]args){
    CheckOdd0r Even();
}
}.
```

COMPLETED
Date: 18/04/2024
Signature:

** Special note.

How to input Char value using Scanner.

```
import.java.util.*;
  Public Class.Example{
    Public Static void main (String[]args){
    Scanner input. = new Scanner (System.in);
      System.Out.print("Input the letter");
      Char ch = input.next().charAt(0);

      System.out.println(ch);
    }
```

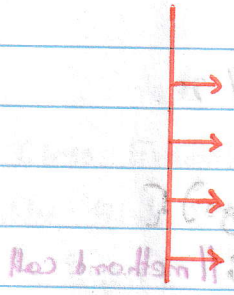
No :

Date: 25/ 04 / 2024

① Java methods

A Java method is a collection of Statement. that perform a specific task and we use java methods is for the increase the our Java program readability..

* There have four method types, they are,



* Parameter Less - void type method.
* Parameterized - void type method.
* Parameter Less - return type method.
* Parameterized - return type method.

② Parameter Less - Void type method.

```
import java.util.*;
Public Class Example {
    Public Static void printInfo(){  // Method declaration. Statement.
        System.out.println("Dinuka");
        System.out.println("20");
    }.
    Public Static void main (String[] args){
        printInfo();  //method Calling Statement.
    }
}.
    // At in there the parameter Count is "0"
    // Argument Count = 0
```

Output.

Dinuka.

20.

③ Parameterize - void type method.

```
import java.util.*;
Public Class Example {
    Public Static void printAge (int age){  // Parameterized method
        System.out.println("The age is :" + age);
    }
```

Output

The age is : 19

No: 25 Date: __/__/____

⊛ Parameterless - return type method

```
import java.util.*;
Public Class Example{
   Public Static int getRandomNumber(){ //method declare
    Random rand = new Random();
    int num = rand.nextInt(100); // 28
    return num;
Public Static void main(String[]args){
    int data = getRandomNumber(); //method call
/* methanata getRandomNumber eke thiyena
   retern karala thiyena num value eka
   main method eke int data valata
   assign wena eka wenne--- */
    System.out.println("The data :"+data);
}
}
```

// Parameter Count - 0
// Argument Count - 0.

Output:
The data : 28

⊛ Parameterized - return type method

```
import java.util.*;
Public class Example{.
   Public Static String getOddOrEven(int num){
      String result = (num%2==0)?"Even":"Odd";
      return result;
   }.
   Public Static void main(String[]args){
      String s = getOddOrEven(6); //method calling
      System.out.println(s);
```

//method declaration

No: _____ Date: __/__/____

Java method overloading

Method overloading in Java is a feture that allows class to have more than one method with the same name. but with different peremeters. and different parameter types.

Example :

```
Public Class Example {
    Public Static void print(){
        System.out.println("My name is Dinuka");
    }

    Public Static void print(int age){
        System.out.println("I am" + age + "years old");
    }

    Public Static void print(String location){
        System.out.println("I'm from" + Location);
    }

    Public Static void main(String[]args){
        print();
        print(20);
        print("Panadura");
    }
}
```

Output.

```
My name is Dinuka.
I am 20 years old.
I am from Panadura
```

Special note.

Method Signature. → Parameter. + method name.

methord Overloding ≠ methord names are equal

No : _____ Date: __/__/____

we cant write java program as like this →

```
Public Class Example.{
    Public Static void Print() {
        System.out.println ("My name is Dinaka");
    }
    Public static void Print() {
        System.out.println ("Im from Bandaragama");
    }
    public static void main(String args[]){
        Print();
        Print();
    }
}.        // Error
```

At this example there method names and parameter counts are also same. then Jvm cannot understand what need to call first. Then there have Compile error.

① Method Calling Stack & Heap.

The Java Stack and heap are two fundamental Concept in Java programming that seve. different purposes.

① Stack memories →

* The stack is a region memory where memory is allocated and dealocated in a Last-In-First-Out order. (LIFO)
* Used to → Store local variables
  method Parameters.
  return addresses.

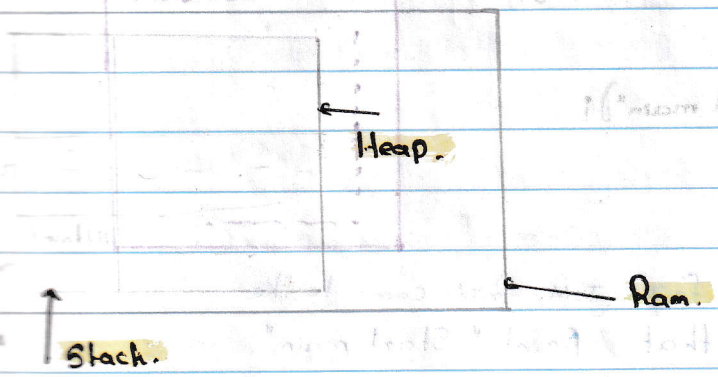
FILO → First In Last Out

No: _____ Date: __/__/____

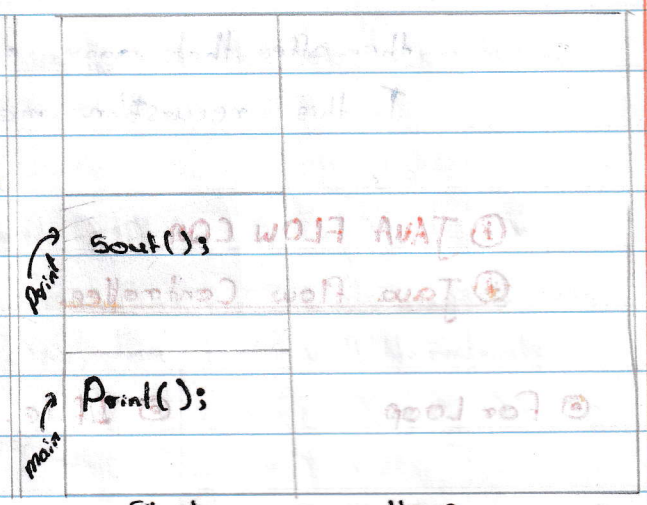
① Heap memory →

- The heap is region of memory where object are allocated and deallocated dynamically.
- It is used to store objects that are created using the new keyword or through deserialization.
- The heap is managed by the JVM, which is responsible for allocating and deallocating memory on the heap.

Special note.




```
Public Class Example {
    Public static void print(){
        System.out.println("My name is Dinuka");
    }
    Public static void main(String[] args){
        print();
    }
}
```



In this Example. first Jvm come to main method. then first stack. mark the main method. after that in main method there have call to print. it in stack after that create print() method in stack. then after that. print "My name is Dinuka" then after that end print method after that firstly Clear the print method on ram after that Jvm come to method called Statement and Continiue . and end main method.

No : _____ Date: __/__/____

① Recursion Method

```
Public Class Main {
    Public Static void recursion() {
        System.out.println("Start recursion()");
        recursion();
        System.out.println("end recursion");
    }
    Public Static void main (String[] args){
        System.out.println("Start main");
        recursion();
        System.out.println("end main");
    }.
}.
```

Outputs

Start main.
Start recursion()
Start recursion()
Start recursion()
.
.
.
.
.
.

In this Java program first JVM first came to the main method then after that print "Start main" after that recursion method after that print "Start-recursion" then after that again run recursion() method. In this recursion method didn't stop, never.

② Java flow Controlles

① For Loop

```
for (int i = 0; i<5; i++){
    |
}
```

① ② ③

① - Variable declare and initialized.
② - Boolean expresson.

② While.

```
while (i<5) {
    |
}
```

①

① - Boolean expresson.

③ do-while.

```
do {
    |
} while (i<5);
```

①

① - Boolean expresson

At in there we

No: _____ Date: __/__/____

① For Loop.

① → variable declare and initialize.
② → Condition.
③ → Increment. / decrement.

* How to for loop work.

```
for (int i = 0 ; i < 5 ; i++) {
    System.out.println("Hello world");
}.
```

① first JVM come to the for loop after that. there have declare and initialization the variable. after that. Jvm come to the Conditon part. at ② there is the condition is true JVM Come to the for loop body. In this example there have S.out "Helloword" after print that for loop frist itteration is end. then again Jvm go to the for Loop Condition part. is it true then ③ before Jvm goto the for loop body increment. the "i" after that Jvm goto the for loop body this is done by for Loop again and again while the condition is true when the Condition is false the for loop be end.

① variable declaration and initialization.

This is where you set the starting value of the loop counter....

② Condition.

This is a Condition that is checked at the begining of each iteration.. If the Condition is true. the Code inside the loop will be executed. if the Condition is false, the loop will terminate.

③ Increment or decrement.

This is where you increment the loop. Count by a certain value.
(i++) / (i--)

→

No: _____ Date: __/__/____

Examples.

example:

```
Public Class Example {
    Public Static void main(String[] args) {
        for(int i=0; i<5; i++){
            System.out.println("Hello world");
        }.
    }
}.
// In this code the for loop iterate the
// five times. and Print. the Hello world
// five times.
```

Output.

Hello world
Hello world
Hello world
Hello world.
Hello world.

* Incrementing for Loop.

```
Public Class Example {
    Public Static void main (String[] args){
        for(int i=5; i>0; i--){
            System.out.println (i);
        }.
    }
}
// In this Java program there print the numbers
what have the number assign to the variable "i"
to the whole condition is true. //
```

* decrementing for Loop

5
4
3
2
1

At in the decriment the "i"

① Java for loop Cases.

Case no.	Example.	Output.

Case I.	Public Class Example{
	Public Static void main(String[] args){
	for (){ // Illegal.
	}
	}
	}
		Error.

No: _____ Date: __/__/____

I
```
Public Class Example {
  Public Static void main (String[] args){
    for ( ; ;) {  // Legal.
    }
  }
}
```

II
```
Public Class Example {
  Public Static void main(String[] args){
    int i =0;
    for ( ;i<5; i++){  // Legal.
      System.out.println( "Hello");
    }.
  }
}.
```

0
1
2
3
4

III
```
Public Class Example {
  Public Static void main(String[] args) {
    for(int i=0 ; i<5 ; ) {   ? i++ / i--
      System.out.println("Hello");  // Legal.
    }
  }
};
```

Hello
Hello
Hello
Hello
Hello.
:
:
Hello.

/* In this Java program. without the increment. the jum Check the condition but anytime. Condition didn't false. then the java program run at Streech. So this is calling infinity Loop */

Infinity Loop → The infinity Loop is mean it's never stop. as like in this example iterate the for loop again by again but never stop

IV
```
Public Class Example {
  Public Static void main (String[] args){
    for (int i=0; i<5 ;) {
      System.out.println("Hello");
```

{Hello
Hello
Hello
Hello

No :_____ Date: __/__/_____

Case vi.

```
Public Class Example{
  Public Static void main (String[] args){
    for(int i=0; i<10;){
      System.out.println("Hello");
    }.
    i++; // Illegal → variable can't find.
  }
}.
```

Error.

Case vii.

```
Public Class Example {
  Public Static void main(String [] args){
    for(int x=0; x<5; x++){
      System.out.println(x);
    }
  }
}.
```

0
1
2
3
4

/* In this Java Program in for Loop use the int x. at in this code is no any error but we use i: for itteration that is the standad in java software engneers */

① While Loop:

* How to work Java while Loop.

```
Public Class Example {
  Public Static void main (String [] args){
    while ( ① ){
    }
  }
}
```

① → Condition.

/* In this Java while loop at in their first Check the Condition. after that ran the while loop body while condition

No :

Date:

Incomplete
Signature:
Date: 25/4/2024
INCOMPLETED

Example. -

Example.

```
Public Class Example {
   Public static void main (String[] args){
      int i = 0;
      while (i<10){
         System.out.println("Hello");
         i++;
         }
   }
```

Output

```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello.
```

INCOMPLETED
Date: 2/5/2024
Signature:

```
Public Class Example {
Public Static void main (String [] args) {
   int i = 5;
    while (i>0){
     System.out.println("i :" + i);
     i--;
     }
   }
}.
```

```
i : 5
i : 4
i : 3
i : 2
i : 1
```

① Do while Loop.

* How to work java do-while loop.

```
Public Class Example {
   Public Static void main (String [] args).{
        do {
           | // This is a body of do while loop.
           } while ( ① );        ① ⟶ Condition
   } /* In this do while loop we have write statements on
}.    do while loop body this is also same to the while loop. */
```

No : _____ Date: __/__/____

Examples.

```
Public Class Example {
    public static void main (String[] args){
        int i = 0;
        do {
            System.out.println ("Helloworld"); i++;
        } while (i<5);
    }
}
```

Output.

```
Hello world
Hello world
Hello world.
Hello world.
Hello world.
```

```
Public Class Example {
    Public static void main (String[] args){
        int i = 0;
        do {
            System.out.println("I : " + i);
            i++;
        } while (i<5);
    }
}.
```

```
I : 0
I : 1
I : 2
I : 3
I : 4
```

```
Public Class Example {
    Public static void main(String[] args){
        int i = 5;
        do{
            System.out.println("Hello");
            i--;
        } while (i>0);
    }
}
```

```
Hello.
Hello.
Hello.
Hello
Hello.
```

```
Public Class Example{
    Public static void main (String[] args){
        int i = 5;
        do{
            System.out.println("i" : " + i);
            i--;
```

```
i : 5
i : 4
i : 3
i : 2
```

No : _____ Date: 02/04/2022

⑧ Nested Loop ← loop inside loop.

* * * * *

* * The nested loop is mean in java there haav a loop statement inside the another loop statement.

* * * * *

```
for (int i=0; i<5 ; i++).{
    for (int j=0; j<3; j++){
    }
}.   /* At in this nested loop the frist for loop
        iterate the second for loop five times and second for loop
        iterate ~~the~~ it self 3 times*/
```

* * * * *

Examples - * *

* * *

Examples. * *

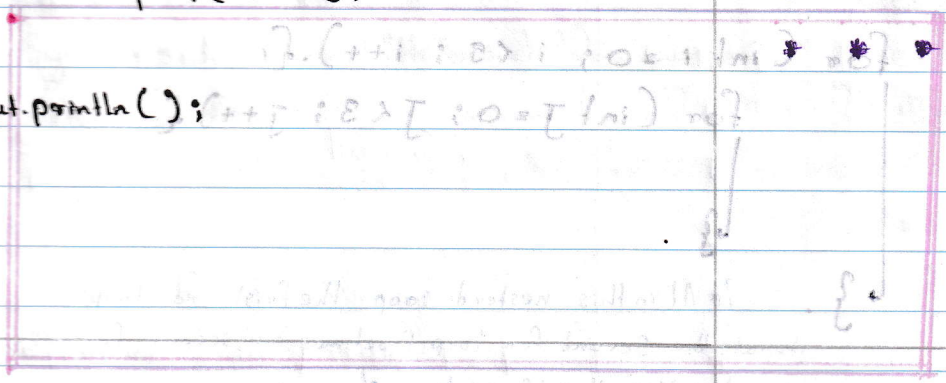
```
public class Example{
  public static void main(String[] args){
    for (int i=1; i<5 ; i++){
      for (int j=0 ; j<i; j++){
        System.out.print("* ");
      }
      System.out.println( ); /new line
    }
  }
}  /* At in this java program frist check. i<5 is it
      true. in this java program that is true. the farist
      for loop come to that's body after that in second
      for loop initialize and declare the variable j.
      after that check. If is j<i is it true go to the
      body then 5 out the start after that end second body
      after that again check the j<i is it true is if false
      then jum come to frist for loop body after that
```

Outputs.

```
*
* *
* * *
* * * *
```

No: 50 Date: __/__/____

```
Public class Example{
   public static void main (String[] args){
      for(int i =0; i<5; i++){
         for (int j=0; j<5 ; j++){
            System.out.print("* ");
         }
        System.out.println();
        }
   }
}.
```



```
public class Example{
   public static void main (String[] args){
      for ( int i = 0; i<5; i++){
         for ( int j = 5 ; j>i ; j--){
            System.out.print ("* ");
         }
        System.out.println ();
      }
   }
}
```

* * * * *
* * * *
* * *
* *
*

⑧ Java Label.

The Java label in for Loop is used for identify a specific Loop. within a nested loop. when we using Java lables. we can break or continue specific loop. according to our requirement. if we break without using Java lables. it in there the jum break the what we are following for loop.

```
Outer : for (int i =0 ; i<5 ; i++){
   inner: for(int j =5 ; j >1 ; j--){
            }
```

No: _____ Date: __/__/____

Example -

```
public class Example {
public static void main(String[] args){
    L1 : for(int i=1; i<3; i++){
        L2 : for(int j=0; j<4; j++){
            System.out.println(i + "  " + j);
            }
        System.out.println();
        }
    }
}
```

Output:

```
1  0
1  1
1  2
1  3

2  0
2  1
2  2
2  3
```

```
public class Example {
public static void main(String args[]){
    L1 : for(int i=1; i<3; i++){
        L2 : for(int j=0; j<4; j++){
            System.out.println(i + "  " + j);
            }
        System.out.println();
        break L1;
        }
    }
}
```

Output:

```
1  0
1  1
1  2
1  3
```

Special note.

* break. →

The break statement is used to terminate (stop) the loop imidietly when we write the break keyword inside Java loop it exit from the loop and continues with

No :_____ Date: __/__/____

Example-

```
for (int i =0 ; i<10 ; i++) {
    if ( i ==5) {
        break;
    }
    System.out.println(i);
}
```

Outputs-
0
1
2
3
4

/* Methandi wenne for Loop eken 10 parak Loop eka iterate karanawa ethanadi "i" 5ta samanada kiyala balanawa, samana nm for loop eka break karanawa---- ethanin passe statemens thiyenawanm for loop eken eliye ewa nan wenawa */

* Continue. →

When we use the Continue keyword for the res of current itteration of a loop, and proceed the next itteration.

Example-

```
for( int x = 0; x<10 ; x++){
    if ( x == 5){
        Continue;
    }
    System.out.println(x);
}
```

Output.
0
1
2
3
4
6
7
8
9

There have skip the num-5-

No: _____ Date: _____

* return ⟶.

× we use return keyword for exit method and return value (if method have return value) to the caller

* when we use return keyword it also terminated (stop) the execution of the method.

Example.

```
-----
-----
for(int x = 0; x < 10; x++){
    System.out.println(x);
    return;
}
-----
-----
```

Output.
0.

⊛ Java Arrays:

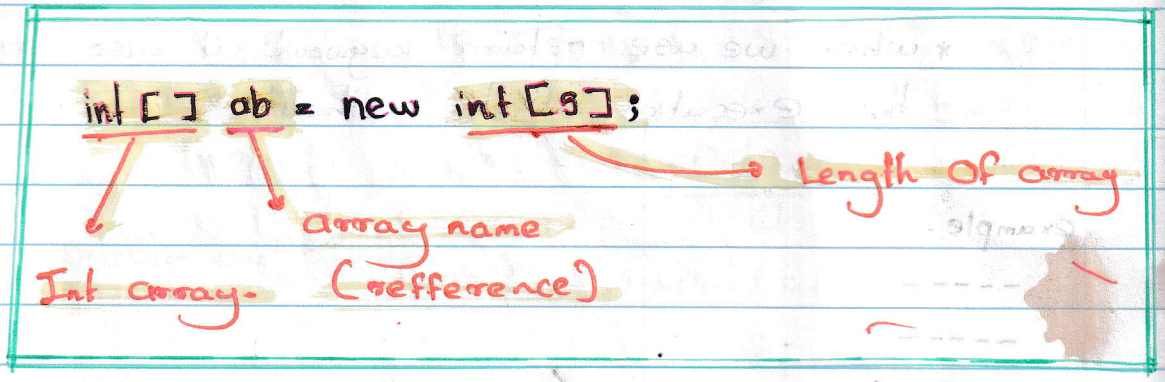
The Java array is mean the Collection of variables of the same datatype using.

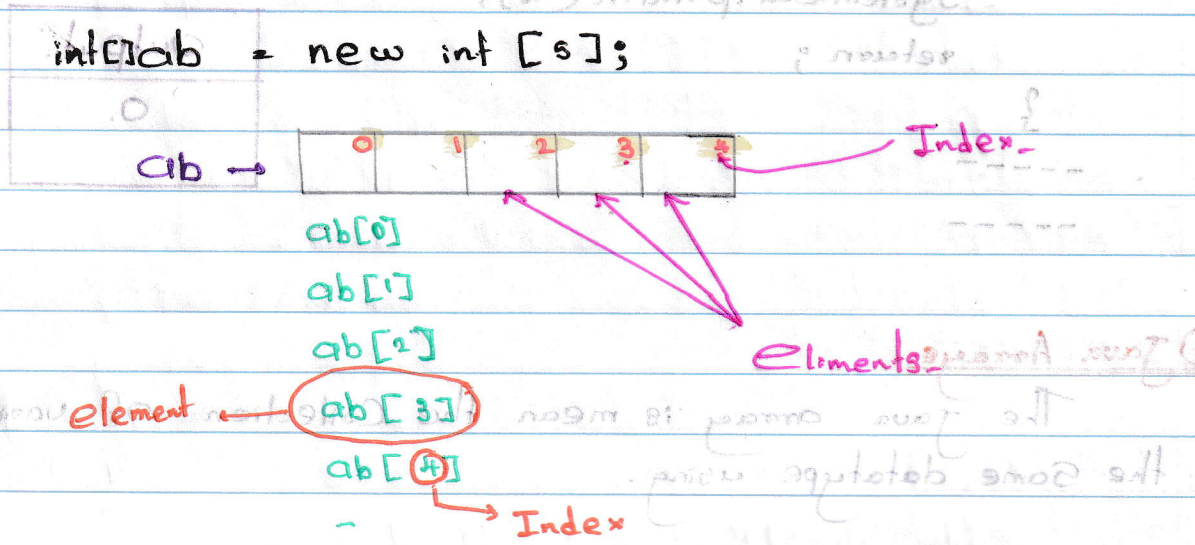
why we use the Java arrays in Programing?

* Efficient memory allocation - Arrays allow for contiguous memory allocation, witch make it easier to access and manupilate the elements.
* Convenient data storage - Arrays provide a simple and efficent way to store a Collection of variables of the same type, makeing it easier to work with large data sets.
* Improved Cord readability. Arrays allows for cleaner and more concise code, as they enable developers to work with collections of data in single variable.
* flexibility - Arrays can use for the store different

No: _____ Date: ___/___/_____

* Easy to iteration. Arrays can be easily iterate by using for loop.





Examples.

Output.

```
import.java.util.Arrays;
Public Class Example{
    Public Static void main (String [] args){
        int [] ab = new int [3];
        System.out.println("ab :" + ab);
        ab[0] = 10;
        ab[1] = 20;
        System.out.println("ab[0]:" + ab[0]);
        System.out.println("ab[1]:" + ab[1]);
```

```
ab : [I@2518a6ad
ab[0]: 10
ab[1] : 20
ab[2] : 0
```

/* At in there point address of the array. where is it store on the volatile memmory (ram) */

No :

Date: / /

```
import java.util.*;
Public class Example {
    Public static void main(String[] args){
        int[] ab = new int [3];
        ab[0] = 10;
        ab[1] = 5;
        ab[2] = 2;
        System.out.println("ab[0]:" + ab[0]);
        System.out.println("ab[1]:" + ab[1]);
        System.out.println("ab[2]:" + ab[2]);
    }
}
```

```
ab[0]: 10
ab[1]: 5
ab[2]: 2
```

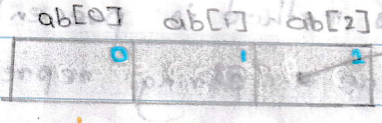
/* At in these two examples. there we have import java.util.Arrays. when we create an array we must need to import array from java. So we can easily use as * "All" import from java files */

④ Java array Length.

When we create the java array at in there have elements we can find howmany ~~Length~~ elements are at there by using .Length.

Special note.

```
int[] ab = new int[3]; // Length = 3.
```



at in this example there have array length is 3. but at in there ② last element index is ab[2]

No: _____ Date: __/__/____

Examples. Output-

```
import java.util.*;
Public Class Example{
  Public static void main(String [] args){
    int[] ab = new int[5];
    System.out.println(ab.Length);
  }
}
```

ab.Length ↑ array name / reference-

Output: 5

```
import.java.util.*;
Public Class Example.{
   Public Static void main(String [] args){
      int[]ab = new int[10];
      //initialization-
      for( int i =0; i < ab.Length ; i++){
            ab[i] = i+1;
      }
      // print-
      for (int x =0 ; x < ab.Length; x++){
         System.out.println("ab["+i+"]:" + ab[i]);
      }
   }
}
```

Output:
ab[0] : 1
ab[1] : 2
ab[2] : 3
ab[3] : 4
ab[4] : 5
ab[5] : 6
ab[6] : 7
ab[7] : 8
ab[8] : 9
ab[9] : 10

Special note

⑧ Arrays.toString

The Arrays.toString() key word is used to Convert an array in to String. This method is precent in java.util.Arrays Class in Java and it returns a String representation of Contents of the specified array.

By using Arrays.toString() we can print all of the

No: _____ Date: __/__/____

Examples.

```
mport. Java. util. *;
Public Class Example{
  Public Static void main(String[] args){
    int[] ab = new int [5];
      for (int i=0; i<ab.length ; i++){
          ab[i] = i+1;
      }
    System.out.println (Arrays.toString (ab));
  }
}
```

Outputs.

```
[1,2,3,4,5]
```

Default data types.

- byte, Short, int, Long → 0.
- double, float → 0.0
- boolean → false
- String → null.
- Char → '\u0000'

Examples.

```
-port java. util. *;
Public Class Example {
  Public static void main (String [] args){
      byte [] bt = new byte [5];
      Short [] st = new Short [5];
      int [] in = new int [5];
      Long [] lo = new Long [5];
      // floting point.
      double [] db = new double [5];
        float [] fl = new float [5];
      // char.
        Char [] ch = new Char [5];
```

Outputs.

```
[0,0,0,0,0]
[0,0,0,0,0]
[0,0,0,0,0]
[0,0,0,0,0]
[0,0,0,0,0]
[0.0,0.0,0.0,0.0,0.0]
[0.0,0.0,0.0,0.0,0.0]
[ , , , , ]
[false, false, false, false, false]
```

No : _____ Date: __/__/____

```
// String
String [] str = new String [5];

System.out.println(Arrays.toString(bt));
System.out.println(Arrays.toString(st));
System.out.println(Arrays.toString(in));
System.out.println(Arrays.toString(lo));
System.out.println( );
System.out.println(Arrays.toString(db));
System.out.println(Arrays.toString(fl));
System.out.println( );
System.out.println(Arrays.toString(ch));
System.out.println( );
System.out.println(Arrays.toString(boo));
System.out.println( );
System.out.println(Arrays.toString(str));
}
}
```

⑤ Arrays declaration rules.

Example-

```
import java.util.*;
Public Class Example {
  Public Static void main (String[]args){
    int []ab = new int[5];
    int [] yr = {10,20,30,40};
    int [] xr = new int[] {100,200,300,400};
    System.out.println(Arrays.toString(ab));
    System.out.println(Arrays.toString(yr));
    System.out.println(Arrays.toString(xr));
```

Outputs-

```
[0,0,0,0,0]
[10,20,30,40]
[100,200,300,400]
```

No :

Date :

Arrays references passing methods.

In java when we pass an array to a method we need to pass the refference (name) of array.

```
Public Static void PrintArray(int [] ab){
    System.out.println(Arrays.toString(ab));
}
Public Static void main (String [] args){
    PrintArray(
    int [] ab = new int [5];
    PrintArray(ab);
}.
```

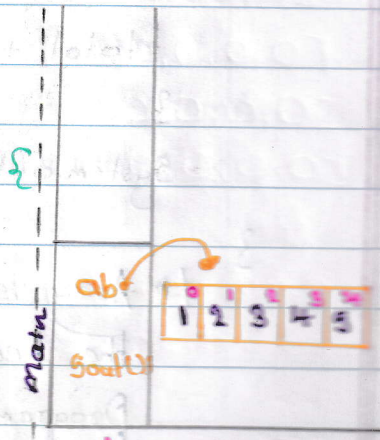
array refference (name).

Special note:

When we Create the array in our Java Program. It Save On our ram tempory in JRE There have a Stack and heap. The java arrays are Save on heap.

Example:

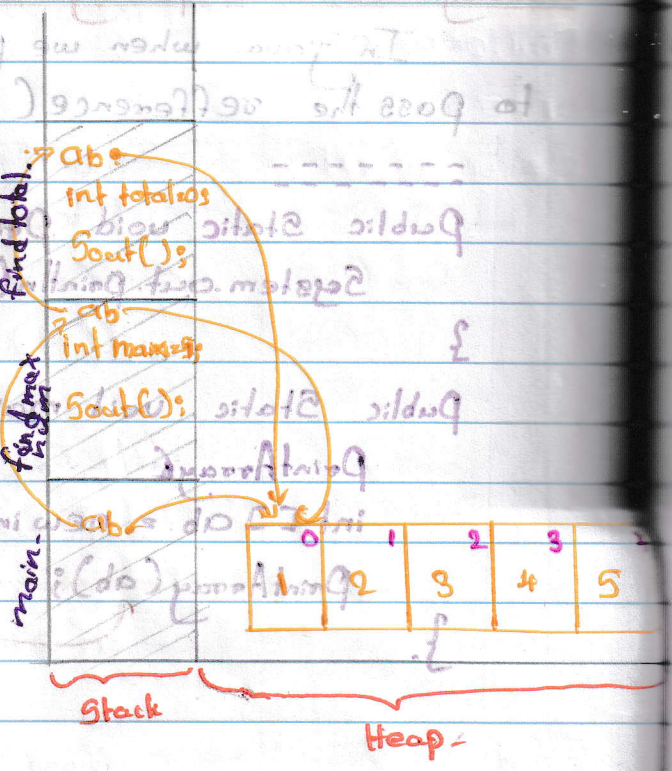
```
import java.util.*;
Public Class Example{
    Public Static void main (String args[]){
        int [] ab = new int [5];
        for(int i=0 ; i<ab.length; i++){
            ab[i] = i+1;
        }
        System.out.println(Arrays.toString(ab));
    }
}
```



Output-

No :_____ Date: __/__/____

```
import java.util.*;
Public Class Example{
  Public Static void main(String[] args){
    int[] ab = new int[5];
    for(int i=0; i<ab.length; i++){
      ab[i] = (i+1);
    }
    Find Max Num (ab);
  }
  Public Static void Find Max Num (int[] ab){
    int max = ab[0];
    for(int i=0; i<ab.length; i++){
      if(max < ab[i]){
        max = ab[i];
      }
    }
    System.out.println("The max num is : " + max);
    find Total (ab);
  }
  Public Static void find Total (int[] ab){
    int total = 0;
    for(int i=0; i<ab.length; i++){
      total += ab[i];
    }
    System.out.println("The Total is : " + total);
  }
}
/* Array is not Erasee from
   the heap while end the
   Program. */
```



Outputs
The max num is: 5
The total is : 15

when we Create the array after that we pass the array for the methods. there didn't make Coppys on heap.

WE CAN ACCESS array ON Any method. we Can Change ~~and~~ array in any method.

No : _____ Date: __/__/_____

Example.

```
import java.util.*;
Public Class Example {
  Public Static void PrintArray(int[] xr){
      System.out.print("[");
      for(int i=0; i<xr.length; i++){
          System.out.print(xr[i] + ", ");
      }
      System.out.println("\b\b]");
  }
  Public Static void main(String[] args){
      int[] xr = new int[] {10,20,30,40,50};
      PrintArray(xr);
  }
}
```

Output

```
[10,20,30,40,50]
```

```
import.java.util.*;
Public Class Example {
  Public Static boolean isLengthEqual(int[] xr, int[] yr){
      return xr.length == yr.length;
  }
  Public Static void main(String[] args){
      int[] xr = new int[5];
      int[] yr = {10,20,30,40,50}

      boolean isLengthEqual = isLengthEqual(xr, yr);
      System.out.println(isLengthEqual);
  }
}
```

True.

No : _____ Date: ___/___/_____

```
import java.util.*;
Public Class Example {
  Public Static void main (String [] args){
    int [] ab = {10,20,30,40};
      PointArray(Ab);
  }
  Public static void pointArray (int[] ab){
      System.out.println(Arrays.toString(ab));
      getTotal(ab);
  }
  Public Static void getTotal( int [] ab) {
      int total =0;
      for(int i=0; i<ab.length; i++){
          total += ab[i];
      }
      System.out.println (" The total is :" +total);
      get Avg (ab);
  }
  Public Static void getAvg (int [] ab){
      int total =0;
      for(int i=0; i< ab.length; i++){
          total += ab[i];
      }
    double avg = (double) total / ab.length;
      System.out.println("The avg is :" + avg);
  }.
}.
```

[10, 20, 30, 40]
The total is : 100
The avg is : 25.0

```
Public Static void getMaxNum(int[] ab){
    int max = ab[0];
      for(int i=0; i< ab.length.; i++){
```

The max num is 50
The min num is 10

No :_____ Date: __/__/_____

```
    System.out.println("The max num is " + max);
    getMinNum(ab);
}

Public static void getMinNum(int[] ab){
    int min = ab[0];
    for(int i=0; i < ab.length ; i++){
        if(min > ab[0]){
            min = ab[0];
        }
    }
    System.out.println("The min num is " + min);
}

Public static void main(String[] args){
    int[] ab = {10,20,30,40,10};
    getMaxNum[ab];
}.
}.
```

```
import java.util.*;
Public Class Example{
    Public Static void printArray(int[] ab){
        System.out.println(Arrays.toString(ab));
        increment(int ab);
    }
    public static void increment(int[] ab){
        for(int i=0; i<ab.length; i++){
            ab[i]++;
        }
    }
    Public Static void increment(int x){
        x++;
    }
Public Static void main(String[] args){
```

[10,20,30,40]
[11,21,31,41]
x : 10
x : 10.

No :_____ Date: ___/___/_____

```
int x = 10;
System.out.println("x :" + x);
increment (x);
System.out.println("x :" + x);
    }
}
```

/* In this example there have increment the array and x in different methods. at in there array have change as the. 1, 2, 3, 4 but x is not change because variable is can't accesss in two methods but we can accsess the array on anyware in Java program */

differents between variable and array

Variable.	Arrays.
* Hold a single value.	* Hold a multiple values as elements.
* Can hold primitive or reference data types.	* Hold elements of same data types.
* Accessed useing its name directly.	* Accessed by its name followed by its index.
* Declare with specific datatype. ex: int num = 5;	* declare with a data type followed by squre brackets. ex: int[] num = {1, 2, 3, 4, 5};

No:_____ Date: __/__/____

① Java runtime errors.

Arrays IndexOutOfBounds Exception.

In java this error is a runtime error. this error have occurs when we tring to accses in index an arrad dosent exist. terms.

Example.

```
import java.util.*;
Public Class Example {
    public static void main (String[] args){
        int [] ab = new int [5];
        System.out.println(ab[6]); //Ahin there have runtime
    }                              error becouse in ab array
}.                                 didn't exist the ab[6];
```

② Java reverse element.

In java array reverse mean flipping the order of elements in an arrays so the frist element becomes the last and second becomes second last. and the array being arranged in the opposite order.

Example. - 1 method. → flip valuse by using another array

```
import java.util.*;
Public Class Example {
  Public static String[] reverse (String[] names){
    String [] temp = new String [names.length];
      for(int i=0; i<names.length ;i++){
        temp[i] = names[names.length-(i+1)];
      }
      return temp;
    }
```

No : _____ Date: ___/___/_____

```
String[] names = {"Dinuka", "Sasani", "Ilma", "Namal"};
System.out.println(Arrays.toString(names));

names = reverse(names);
System.out.println(Arrays.toString(names));
    }
}
```

Output-
[Dinuka, Sasani, Ilma, Namal]
[Namal, Ilma, Sasani, Dinuka]

/* At in this example we have Create the new temp array and add values for it there after that we have return the temp and in main method we have print it *

2.method → flip values by using variable.

```
import java.util.*;
Public Class Example{
public static void reverse (String [] ab) {
    String temp;
    int j = names.length -1;
        for(int i =0; i < names.Length /2 ; i++){
            temp = names [j];
            names[j] = names[i];
            names[i] = temp ;
            j--;
    }
}
public static void main(String [] args){
    String[] names = {"Namal", "Sasani", "Ilma", "Dinuka"};
    System.out.println(Arrays.toString (names));
    reverse (names);
    System.out.println(Arrays.toString (names));
    }
}
```

Output-
[Namal, Sasani, Ilma, Dinuka]

No :

Date: 09/03/2024.

⑧ For each / enhanced loop-

The for each loop in Java used to iterate through each element of an array or a Collection it also known as the enhanced for Loop. The Syntax for a for-each loop is as follows-

```
for (data-type-variable : array name){
    // Code to be executed-
}
```

/* At in these doing same thing as like the ⑦ in for Loop- by use this for each loop we can increase our Java program Simplicity and the readability. At in for each Loop we dont need manage the index variables manualy as like in for Loop when we iterating the arrays or Collections. */

Examples-

```
Public Class Example{
    Public Static void main (String[] args){
        int[] nums = {54, 94, 67, 32, 99, 89, 7};
        for(int i=0; i<nums.Length; i++){
            System.out.println(nums[i]);
        }                    //by using for Loop.
    }
    /* At in there I have get outputs from the array
    line by Line by using the for Loop.*/
}
```

Outputs-

```
54
94
67
32
99
89
7
```

```
Public Class Example{
    Public Static void main (String[] args){
        int[] nums = {54, 94, 67, 32, 99, 89, 7};
        for(int num : nums){
            System.out.println(num);
        }                    //by using for each Loop-
    }
    /* At in there I have write the code using for each
```

```
54
94
67
32
99
89
7
```

No: _____ Date: __/__/____

⑧ Array grow method.

we can grow array length by using grow method we need to create the array grow method

```
Public Class Example {
  Public Static void main (String [] args){
    int [] ab = new int [0];
      for(int i=0; i<5 ; i++){
        System.out.println(ab.length);
      ab=growArray(ab);
      }
  }
  Public Static int[] growArray (int[]ab){
    //array declare here
at in there we ⟶  int [] temp  = new int[ab.length +1];
have declare the  //coppy values here.
(ab.length+1)       for(int i =0 ; i< ab.length ; i++){
                      temp[i] = ab[i];
                    }
                                                  } and after that we have coppy the values for the temp
      return temp;
    }
  }
```

Output
0
1
2
3
4

No: _____ Date: __/__/____

① Array Sorting

what is the meaning of Array sorting.?

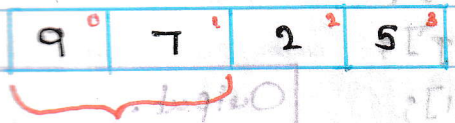
Array sorting is the process of arraing elements of an array in specific order based on a comparison operator This comparison operator. This comparison operator is used to deside the new order of elements in the respective data structure. Sorting can be done in acending or decending order. There have so many algorithems for sort the array. They are;

* Bubble sort
* Selection sort
* Insertion sort.
* Merge sort.

Special note.

How to work the bubble sort.

when we sort the array by ~~sfuns~~ using bubble sort frist we get the frist two elements and after that we check what is the greater value in frist two elements and if the frist element is greater than to the ~~frs~~ second element we switch it. (frist one is go to the second element Pace and second one is come to the frist place.).
After that we check the second one and third element. after that we do that process what we have done at fristelemet. while the array make as the ~~according~~ acending or decending order.



① frist get the frist two elements.

② After that we check as like this.

ab[0] > ab[1]

(# frist element ~~is~~ do second element

No : _____ Date: __/__/____

7 ⁰	2 ¹	9 ²	5 ³

③ After that we get the second element and third element and do same process.

ab[1] > ab[2]

7 ⁰	2 ¹	5 ²	9 ³

There we have get biggest number to the last element.

④ we need to do this while array Sort Correctly.

2 ⁰	5 ¹	7 ²	9 ³

finaly we can get sorted array. as this.

Special note.

Code Example →

```
// array Sort ascending Order.
Import java.util.*;
Public Class Example{
    Public Static void main(String [] args){
    int [] ab = {9,7,2,5};
     System.out.println("array before sort:" + Arrays.toString(ab));
     for(int i = 0; i<ab.length ; i++){
        for(int j = 0; j<ab.length-1; j++) {
          if( ab[j] > ab[j+1]){
            int temp = ab[j];
              ab[j] = ab[j+1];
              ab[j+1] = temp;
            }
        }
    }
```

Output.

array before sort :[9,7,2,5]

array after sort :[2,5,7,

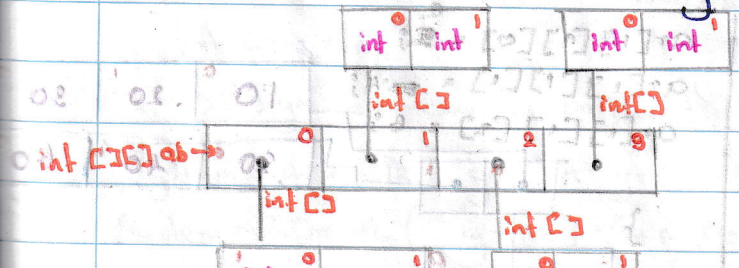
No :_____ Date: __/__/____

```
//array Sort decending order.
Import Java.util.*;
Public class Example{
  Public static void main(String[] args){
    int [] ab = { 2, 7, 9, 5};
     System.out.println ("array before Sort :" + Arrays.toString(ab));
     for(int i = 0; i<ab.length; i++){
       for(int j =0; j<ab.length-1; j++){
           if(ab[j] < ab[j+1]){
              int temp = ab[j];
              ab[j]  = ab[j+1];
              ab[j+1] = temp;
            }
       }System
     }
     System.out.println("Array after Sort:" + Arrays.to String (ab));
  }
}
```

Output.
Array before Sort : [2,7,9,5]
Array after Sort : [9,7,5,2]

① Multi Diamentional Arrays.

Basicaly we use the mully dimensin arrays are used if we want to add arrays inside an array.



In this example, you think we have 4 students are in the class room and they all have done 2 subjects

No :_____ Date: __/__/____

Example →

```
Import Java.util.*;
Public Class Example{
   Public static void main (String [] args){
      // 1 Dimensional array declare
      int [] ab = new int [4];
      // array & initialization
      ab [0] = 10;
      ab [1] = 20;
      ab [2] = 30;
      ab [3] = 40;
```

10 (0)	20 (1)	30 (2)	40 (3)

```
      // 2D array declaration
      int [][] array = new int [2][3];
      // 2D array initialization
      array[0][0] = 10;
      array [0] [1] = 20;
      array[0] [2] = 30;
      array [1] [0] = 5;
      array [1] [1] = 10;
      array [1] [2] = 15;
```

	0	1	2
0	10	20	30
1	5	10	15

(0,0) (0,1) (0,2) (1,0) (1,1) (1,2)

```
      // 3d array declaration
      int [][][] ar = new int [2][2][3];
      // 3d array initialization
      ar [0][0][0] = 10;
      ar [0][0][1] = 20;
      ar [0][0][2] = 30;
      ar [0][1][0] = 20;
      ar[0][1][1] = 40;
      ar[0][1][2] = 60;
      ar[1][0][0] = 8;
      ar[1][1][1] = 4;
      ar[1][1][2] = 6;
   }
}
```

10 (0)	20 (1)	30 (2)

20 (0)	40 (1)	60 (2)

No : _____ Date: __/__/____

Special note.

we Can get the all array output by using Deep to String keyword.

```
System.out.println(Arrays.deepToString(array_name));
```

Example.

```
Import .java.util.*;
Public Class Example{
public Static void main(String [] args){
    int [][] ab = {{10,20,30},{5,10,15}};
    System.out.println(Arrays.deepToString(ab));
}
}
```

Output.
[[10,20,30],[5,10,15]]

Examples.

Example.

Output.

Case I

```
Import Java.util.*;
public Class Example{
  public Static void main(String args[]){
    int [] xr = new int[5];
    System.out.println("xr" + xr);
    System.out.println("xr[0]" + xr[0]);

    int [][] yr = new int[3][2];
    System.out.println("yr:" + yr);
    System.out.println("yr[0]:" +yr[0]);
```

// int there print the address where store at the heap

Print the 0 value (default value).

```
xr : @10...
xr[0] : 0
yr : @1010...
yr[0] : @...
yr[0][0] : 0.
```

No : _____ Date: __/__/____

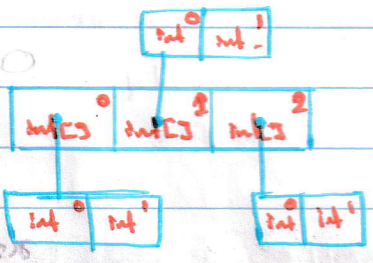
Case II

```
Import java.util *;
Public Class Example{
  Public Static void main (String []args){
     int [] xr = new int [5];
     System.out.println("xr.length :" +xr.length);
     System.out.println("xr[0].Length" + xr[0].length); //Illegal
     int [][] yr = new int [3][2];
     System.out.println("yr.Length:" + yr.length);
     System.out.println("yr[0].Length:" + yr[0].length);
     System.out.println("yr[0][0].Length:" + yr[0][0].Length); //Illegal
  }
}
```

xr: 5
Error
Yr: 3
Yr: 2
Error
//Illegal
//Illegal

Case II

```
Public Class Example{
  Public Static void main (String []args){
     int [][] ab = new int [3][2]
        // Tabular format.
        //Rows → 3
        //Colums → 2
  }
}
```



	0	1	2
0	(0,0)	(0,1)	(0,2)
1	(1,0)	(1,1)	(1,2)

Case IV

```
Public Class Example {
  Public Static void main (String []args){
     int [][] xr = new int [3][];
        System.out.println(xr[0]);
        System.out.println(xr[1]);
        System.out.println(xr[2]);
        int [] yr = null;
```

null
null
null
yr: null

No: _____ Date: ___/___/_____

case v.

```
Import java.util.*;
Public Class Example {
   Public Static void main (String [] args){
     int [][] xr = new int [3][];
     xr[0] = new int[10];
     xr[1] = new int[8];
     xr[2] = new int[2];
     System.out.println("xr[0] length. : " + xr[0].length);
     System.out.println("xr[1] Length : " + xr[1].length);
     System.out.println("xr[2] Length : " + xr[2].length);
   }
}
```

```
xr[0].Length : 10
xr[1] Length : 8
xr[2] Length : 2
```

case vI

```
Import java.util.*;
Public Class Example {
   Public Static void main(String [] args){
     int [] xr = {10, 20, 30, 40};
     int [][] yr = {{10,20,30},{100,200},{1,2,3,4}};
     System.out.println("yr.length : " + yr.length);
     System.out.println("yr[0].Length : " + yr[0].length);
     System.out.println("yr[1].length : " + yr[1].length);
     System.out.println("yr[2].Length : " + yr[2].length);
   }
}
```

```
yr.Length : 3
yr[0].Length : 3
yr[1].Length : 2
yr[2].Length : 4
```

case vII

```
import java.util.*;
Public Class Example{
   Public static void main (String [] args) {
     int [][] yr = {new int [5], new int[10], new int[0]};
     System.out.println("yr.length :" + yr.length);
     System.out.println("yr[0].length : " + yr[0].length);
```

```
yr.length : 3
yr[0].length : 5
yr[1].length : 10
yr[2].Lenth : 0
```

No :_____ Date: __/__/_____

⑧ Nested for-each Loop.

```
import java.util.*;
public Class Example{
   public static void main(String[]args){
      int [][] yr = {{10,20,30},{5,10,15},{1,2,3,4}};
      //Nested for each Loop.
      for(int [] y : yr){
      ¦ for(int q : y){
      ¦ ¦ System.out.println(q);
      ¦ }
      ¦
      ¦  System.out.println();
      ¦ } }
   }
}
```

Output.
10
20
30
5
10
15
⋮

~ *** ~