

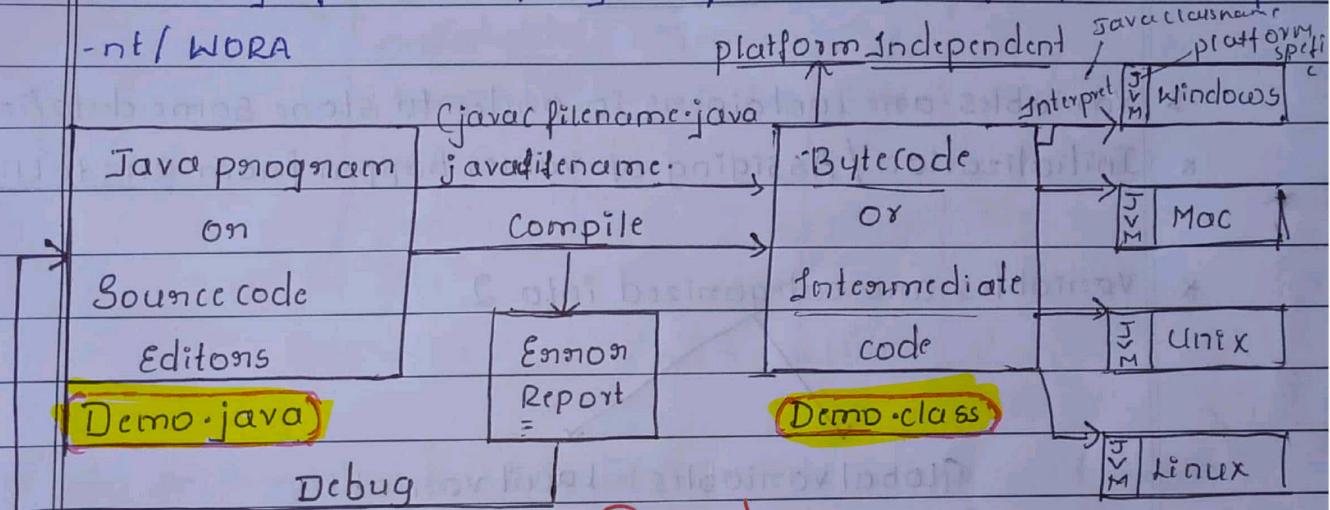
Java operators

Java - Types of operators

Java

- * Java is a high level programming language.
- * Programming Language is a medium to interact with System.
- * Highlevel language is a language in a normal English i.e. Human understandable form.
- * James Gosling was a person who introduced Java.
- * The company which started java is SunMicrosystems (system).
- * Currently Java is owned by oracle.

Working of a Java Program / How is java platform independent / WORA



@codees_notes

JVM : Java virtual Machine (Platform independent)

WORA : Write Once Run Anywhere

Bytecode is an intermediate which is neither low-level nor high-level language so it uses jvm → ① convert to machine level
② Execute line by line.

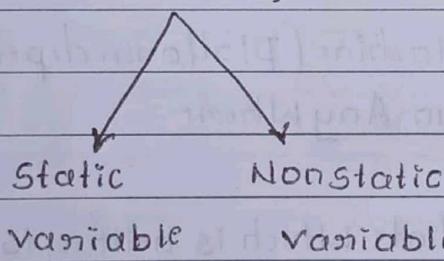
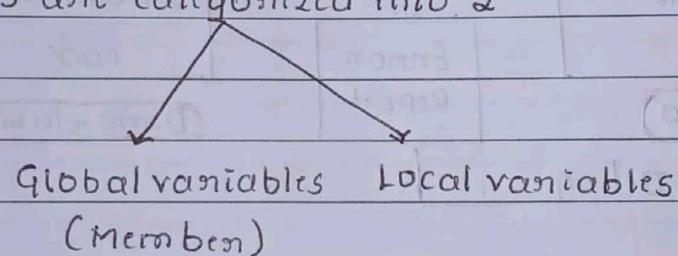
* Firstly we build the java program using editors and save it with the extension .java

* Once we are done developing the program we need to compile it.

- * compilation is a process in order to check if there are any errors in my java program or not
- * If compilation is unsuccessful we get error report based on error report we need to debug the program
- * If compilation is successful we generate bytecode which is intermediate code / platform independent code
- * Extension of all bytecode is **class**
- * This bytecode can be executed on all platform i.e all operating systems

Variables

- * Variables are containers in order to store some data/information
- * Initialization / Assigning of values happen from RHS to LHS
- * Variables are categorized into 2



@codees_notes

Example : Age = 25

LHS	RHS	25
-----	-----	----

Age

Height = 5.5

5.5

Height

V > LHS RHS

Datatypes

@codees_notes

- * Datatypes are used to indicate or specify the type of data stored into variables
- * Datatypes are categorized into 2
 - 1) primitive Datatype
 - 2) Non primitive Datatype
- * In order to store non decimal numeric values we have the following datatypes
- * The difference between those datatypes is Memory size.

Data types	Memory Size	
	Bytes	Bits
byte	1	8
short	2	16
int	4	32
long	8	64

- * In order to store decimal numeric values, we have the following datatypes

Data-T	Memory Size	
float	4	32
double	8	64

widely used and [↑] default datatype for decimal.

- * In order to store true / false we have following datatype

Data-T	Memory size	
	Bytes	Bits
Boolean	1	8

@codees_notes

- * In order to store a single character, we make use of char
- * char data should be enclosed within 'single quotes' ''
- * The Memory size of char is 2 bytes = 16 bits

Note : All the above mentioned 8 data types are together called as primitive data types.

String : It is a data type to store a sequence of characters.

* String data has to be enclosed within "double quotes"

Note : Java is Case Sensitive where in lowercase letters are not equivalent to uppercase letters/values (a ≠ A)

22/01/2020

Variable Declaration

Syntax:

```
datatype VariableName;  
int age;  
double salary;
```

@codees_notes

Variable Initialisation

Syntax:

```
VariableName = value;
```

```
age = 20;           | 20 |  
                    | Age |  
Salary = 5000.69 | 5000.69 |  
                  | Salary |
```

Variable Declaration & Initialization

Syntax:

Initialization

```
datatype VariableName = Value;
```

Declaration

```
boolean x = true;
```

false;

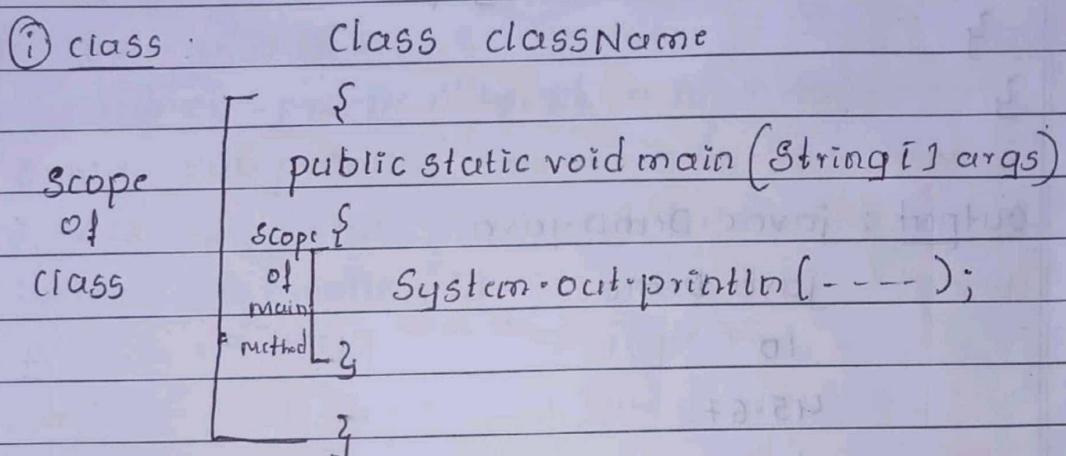
String subject = "java";
= "LJSP2020",
char gender = 'M';

(String can store number,
characters but it should
be written in " quotes")

Structure of a java program.

- 1) Class
- 2) Main method
- 3) Print statement

@codees_notes



The filename should be same as classname during file saving className.java

Ex:1) class Firstprogram

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

```
    System.out.println ("Hello world");  
}
```

@codees_notes

Output: Open cmd

cd desktop

cd java programs

javac Firstprogram.java - (compile) + to enter to

java Firstprogram

- (interpret) the save files

Hello world!!!

Ex:2 Class Demo

{

```
public static void main (String [] args)
```

{

```
System.out.println (10);
```

```
System.out.println (45.67);
```

```
System.out.println (true);
```

```
System.out.println ('Z');
```

```
System.out.println (" Ijsp@2020");
```

y

y

Output : javac Demo.java

```
java Demo
```

```
10
```

```
45.67
```

```
true
```

```
Z
```

```
Ijsp@2020
```

@codees_notes

Ex3) Write a java program to follow the below statement
/ Scenarios

i) Create class called as Student.

ii) Define main method

iii) Under main method initialize 2 variables called as name and age entering those respective values

→ class Student

{

```
public static void main (String [] args)
```

{

```
String name = "Bhagya";
```

```
int age = 23;
```

```
System.out.println (name);
```

```
System.out.println (age);
```

O/P

Bhagya

23

Notes: In java, in order to perform concatenation we make use of 't' operation.

Q) class Employee

{

 public static void main (String [] args)

{

 int id = 101;

 String name = "Jerry";

 double salary = 123.45;

 System.out.println ("Employee Id : " + id);

 System.out.println ("Employee Name is : " + name);

 System.out.println ("Employee Salary = " + salary);

 System.out.println (id + " " + name + " " + salary);

}

Output

@codees_notes

javac Employee.java

java Employee

Employee Id=101

Employee Name is : Jerry

Employee Salary= 123.45

101 Jerry 123.45

23/11/2020

Operators

- 1) Arithmetic Operators
- 2) Assignment Operators
- 3) Relational / Conditional / Comparison Operators
- 4) Logical Operators
- 5) Unary Operators

1) Arithmetic Operators

+ : Addition

- : Subtraction

* : Multiplication

/ : Division

% : Modulus

5 → division

90

10

0 → modulus

Ex: $10/2=5$

$10 \% 2 = 0$

Example:

Class Arithmetic Operators

{

public static void main (String[] args)

{

int x=10;

int y=20;

int sum=x+y;

int diff=x-y;

System.out.println ("Sum = " + sum);

System.out.println ("Difference is " + diff);

System.out.println (y*5);

System.out.println (30/3);

System.out.println (30 % 3);

Output:-

Sum = 30

Difference is = -10

100

10

0

@codees_notes

2) Assignment Operations

=

+ =

- =

* =

/ =

% =

@codees_notes

int $\downarrow a = 5;$ a 5

$a = a + 10$ or $a += 10$ a 15

$a = 15$

int $x = 30$ x 30

$x = 20$ x 20

$\downarrow x = x - 20$ x 10

$x = 30 - 20$ x 10

$x = 10$

Ex: Class Assignment Operation

{

public static void main (String [] args)

{

int $x = 10;$

System.out.println ("value of x is :" + x);

$x += 20;$

System.out.println ("value of x is :" + x);

System.out.println ("= = = = =");

int $a = 6$

System.out.println ("value of a is :" + a);

$a *= 5;$

System.out.println ("value of a is :" + a);

3

QIP

Output

Value of x is 10

value of x is 30

= = = = =

value of a is 6

value of a is 30

3) Relational/conditional /comparison Operators

< : less than

> : Greater than

<= : less than or equal to

>= : Greater than or equal to

== : Equals to

!= : not equal to

Note: Comparison Operations will always return boolean values i.e (true/false)

Ex: Class comparison operation

```
public static void main (String[] args)
{
    int x=10;
    int y=20;
```

@codees_notes

```
boolean result1 = x < y;
```

```
boolean result2 = y > x;
```

```
System.out.println(result1 + "\t" + result2);
```

```
System.out.println (" == != <= >= ");
```

```
System.out.println ( x <= 10 );
```

```
System.out.println ( 3 >= 4 );
```

```
System.out.println (" --- ");
```

```
System.out.println ( x == 10 );
```

```
System.out.println ( y == 30 );
```

```
System.out.println (" --- ");
```

```
System.out.println ( x != 10 );
```

```
System.out.println ( y != 30 );
```

D/P

true true

false

true

false

false

true

4) Logical operations

AND $\Rightarrow \&&$
 OR $\Rightarrow ||$
 NOT $\Rightarrow !$

} return \Rightarrow boolean
 true = 1
 false = 0

AND		$\&\&$		OR		$ $		NOT!	
T	T	T		T	T	T		T	F
T	F	F		T	F	T		F	T
F	T	F		F	T	T			
F	F	F		F	F	F			

Ex: class logical operations

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

```
{
```

```
int x=10;
```

```
int y=20;
```

```
boolean result1 = x < y && y > x;
```

```
boolean result2 = x < y && x == 1;
```

```
System.out.println(result1);
```

```
System.out.println(result2);
```

```
System.out.println(" --- ");
```

```
System.out.println(1 < 2 || 2 > 10). @codees_notes
```

```
System.out.println(2 < 1 || 2 == 3);
```

```
System.out.println(" --- ");
```

```
System.out.println(!true);
```

O/P
true

```
System.out.println(!false);
```

false

```
System.out.println(" --- ");
```

```
System.out.println(!(1 < 2));
```

y
y

true

false

@codees_notes

false

true

false

5) Unary Operators

++ (increment by 1)
-- (Decrement by 1)

++ → pre increment
++ → post increment
-- → pre decrement
-- → post decrement

Ex : Class Unary

{

public static void main (String [] args)

{

int x = 5;

System.out.println ("x:" + x);

x++;

System.out.println ("x:" + x);

++x;

System.out.println ("x:" + x);

x--;

System.out.println ("x:" + x);

--x;

System.out.println ("x:" + x);

}

y

@codees_notes

Output ⇒ x:5

x:6

x:7

x:6

x:5

① int x = 10;

int y = x++;

post increment

↓
First assign, then increment

x [10]

y [10]

② int a = 5

int b = ++a;

pre-increment

↓
First increment, then assign

a [6]

b [6]

③ int i=2;
j=i--;
post-decrement
First assign, then decrement
i [2] j [2]

④ int a=5;
int b=a++;
preincrement
First increment, then assign

⑤ int p=50;
int q=--p;
pre-decrement
First decrement, then assign
p [50] q [49]

@codees_notes

Ex) class Unary

{

 public static void main(String[] args)

 {

 int q=123;

 int w=q+i;

 System.out.println(q+" "+w);

 System.out.println(a+---+n);

 int o=444;

 int p=--o;

 System.out.println(o+" "+p);

3

3

@codees_notes

Decision / conditional statements

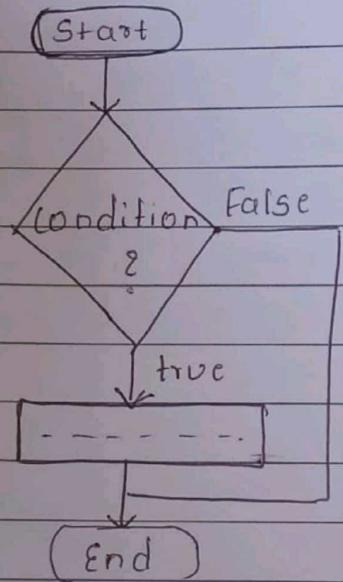
These statements are used to take some decision based on the condition specify. The different decision statements are as follows

- 1) Simple if
- 2) if else
- 3) if else if
- 4) Nested if
- 5) Switch

① Simpleif

Simple if is a decision making statements wherein we execute a set of instructions only if the condition is true

Ex :- Flow chart



Syntax :

```
if (condition)
{
    Statement 1 ;
    - - - - -
    Statement n ;
}
```

Ex :- class SimpleifDemo

@codees_notes

```
{  
    public static void main (String [] args)  
    {  
        System.out.println ("START");  
        int a=10;  
        int b=10;  
    }  
}
```

if ($a \leq b$)

{

System.out.println ("at " is less than or equal to " + b);
}

}

System.out.println ("End");

}

}

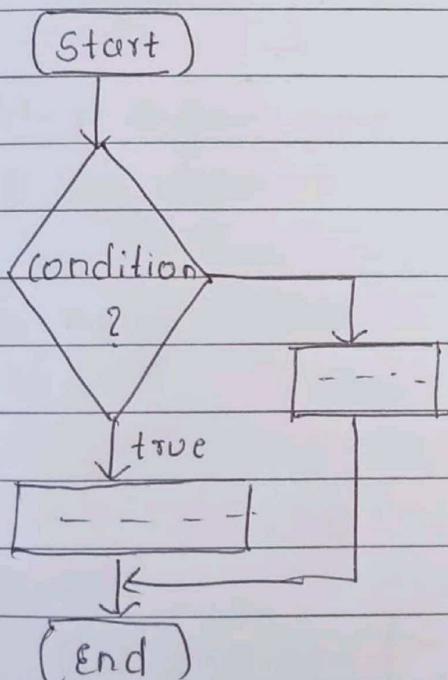
② if else:-

@codees_notes

x if else is a decision making statement wherein, if the condition is true we execute if block. Otherwise if the condition is false we execute else block:-

flow chart

Syntax:



if (condition)

{

Statement 1;

- - -

Statement n;

} set of
instructions

}

else

{

- - -

}

Ex

class IfElseDemo

{

 public static void main(String[] args)

{

 System.out.println("START");

 int x = 100;

 if (x <= 10)

{

 System.out.println(x + " is lesser than or equal to 10");

}

 else

{

 System.out.println(x + " is greater than 10");

}

 System.out.println(" ----- ");

 if (true)

{

 System.out.println(" HI ");

}

 else

{

@codees_notes

 System.out.println(" BYE ");

}

 System.out.println(" ----- ");

*

 if (false)

{

 System.out.println(" WELCOME ");

}

```
else
{
    System.out.println("Thank you");
}
System.out.println("END");
}
```

Output: START
100 is greater than 10

Hi

Thankyou
END

Q) Write a java program to find a number is +ve or -ve
→ Class Number

```
public class Number
{
    public static void main(String[] args)
    {
        int x = 5;
        if (x > 0)
        {
            System.out.println("x is a positive number");
        }
        else
        {
            System.out.println("x is a negative number");
        }
    }
}
```

@codees_notes

Q) Write a java program to find a number is even or odd

```
→ class Number1
{
    public static void main(String[] args)
    {
        int num=4;
        if (num % 2 == 0);
        {
            System.out.println("num is a even number");
        }
        else
        {
            System.out.println("num is a odd number");
        }
    }
}
```

@codees_notes

3) Write a java program to find maximum of 2 numbers

```
→ class Number2
{
    public static void main(String[] args)
    {
        int x=5;
        int y=10;

        if (x > y)
        {
            System.out.println(x + " is a larger than " + y);
        }
        else
        {
            System.out.println(x + " is a less than " + y);
        }
    }
}
```

@codees_notes

if - else - if

if - else - if is a decision making statement where in we can check multiple conditions

Syntax: if (condition)

{

--

}

else if (condition)

{

}

else if (condition)

{

[]

}

else

{

optional

[]

}

Ex.: class IfElseIfDemo

{

psvm (String[] args)

{

if num = 250;

@codees_notes

if (num <= 10)

{

System.out.println ("num + " is less than or equal to 10");

}

```
else if (num <= 20)
{
    s-o.println("numt " is lesser than or equal to 20");
}
else if (num <= 30)
{
    s-o.println("numt " is lesser than or equal to 30");
}
else
{
    s-o.println("Above conditions did not match");
}
```

@codees_notes

O/P: Above Condition did not match.

0-34-F
35-59-F
60-100-F

EX class IfElseIfDemo1

```
s
public class IfElseIfDemo1
{
    int marks=25;
    if(marks >=0 && marks <=34)
    {
        s-o.println("fail");
    }
    else if (marks >=35 && marks <=59)
    {
        s-o.println("First class");
    }
}
```

```
elseif (marks >= 60 && marks <= 100)
```

```
{  
    s.o.pln("FCD");
```

y

else

```
{  
    s.o.pln("Invalid Marks");
```

y

@codees_notes

y

y

O/P : Fail

(4)

Nested if:

Nested if is a decision making statement where in, one if is presented inside another if condition

Syntax: if (condition)

```
{  
    if (condition)
```

```
    {  
        ---  
        --
```

y

else

```
{  
    }
```

Ex ① class Nested If Demo

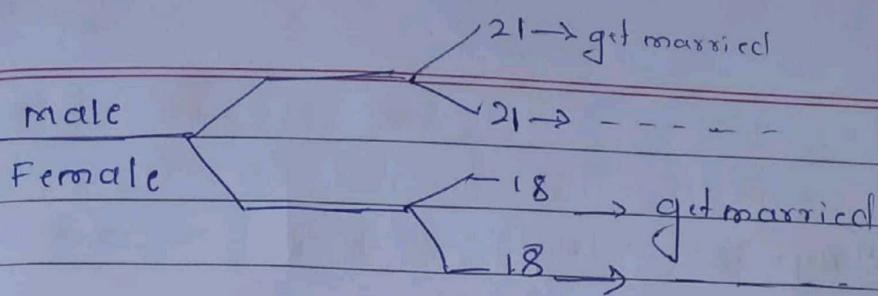
```
class NestedIfDemo
{
    public static void main (String [ ] args)
    {
        char id = 'b';
        int password = 123;

        if (id == 'a')
        {
            System.out.println ("User id is valid");
            if (password == 123)
            {
                System.out.println ("password is valid");
                System.out.println ("Login is successful");
            }
            else
            {
                System.out.println ("Password is invalid");
                System.out.println ("Login is unsuccessful");
            }
        }
        else
        {
            System.out.println ("User id is Invalid");
            System.out.println ("Login is Unsuccessful");
        }
    }
}
```

Output :- ① User id is Invalid

 Login is Unsuccessful

② User id is valid
 Password is valid
 Login is successful

Ex

class NestedIfDemo1

{

PSVM (String [] args)

{

char gender = 'M';
int age = 24;

if (gender == 'M');

{

System.out.println ("Male");

if (age >= 21)

{

System.out.println ("Age is :" + age);

System.out.println ("Get married & hopefully stay happy");

}

else

{

System.out.println ("Age is :" + age);

System.out.println ("Have patience");

}

else if (gender == 'F')

{

System.out.println ("Female");

if (age >= 18)

{

System.out.println ("Age is :" + age);

}

@codees_notes

else

{

s.o.println("Age is :" + age);
()

}

2

else

{

s.o.println("Gender is Invalid");

}

3

O/P :- Gender is Male

age : 21

Get married & hopefully stay happy

stay/happy

② Gender is Invalid

Q) Write a java program to find largest of 3 numbers.

→ class LargestOfThreeNumbers

{

psvm(String[] args)

{

int a = 10;

int b = 5;

int c = 3;

s.o.println("a :" + a + " b :" + b + " c :" + c);

if (a > b)

{

if (a > c)

{

s.o.println("a is largest");

}

@codees_notes

else

{

 s.o.pn("c is largest");

}

 else if (b > c)

{

 s.o.pn("b is largest");

}

 else if (c > b)

{

 s.o.pn("c is largest");

}

 else

{

 s.o.pn("invalid");

}

}

 s.o.pn(" - - - ");

@codees_notes

if (a > b && a > c)

{

 s.o.pn("a is largest");

}

else

{

 s.o.pn("invalid")

}

 else if (b > a && b > c)

{

 s.o.pn("b is largest");

}

 y

 else if (c > a && c > b)

{

 s.o.pn("c is largest");

y

⑤ Switch statement :-
Switch is a conditional statement generally used for character comparison

Syntax :- switch (choice/input)
{

case 1 : - - - -
 break;

Case 2 : - - - -
 break;

- - -
- - -

@codees_notes

Case n : - - -
default :

Ex. class switchDemo

```
class switchDemo  
{  
    public static void main(String[] args)  
    {  
        int choice=3;  
        switch(choice)  
        {  
            case 1 : System.out.println("In case 1");  
                break;  
            case 2 : System.out.println("In case 2");  
                break;  
            case 3 : System.out.println("In case 3");  
                break;  
            default : System.out.println("Invalid choice");  
        }  
    }  
}
```

Output :- In case 3

Break: is a keyword which is used to transfer the control outside the currently executing block

Ex. class monthvalidation

```
public class monthvalidation {
    public static void main(String[] args) {
        System.out.println("start");
        char month = 'Z';
        switch(month) {
            case 'J': System.out.println("In January");
                break;
            case 'F': System.out.println("In February");
                break;
            case 'M': System.out.println("In March");
                break;
            default: System.out.println("Invalid month");
        }
        System.out.println("End");
    }
}
```

@codees_notes

Output: start

Invalid Month

End