Java . Sunbeam (Java foundation)

JDK Installation

Java History & standards

Class and objects

Constructor, destructor, getter, setter

Scanner class

static keyword, static field, static member function.

User defined functions

Package concepts

Array

Inheritance

strings

Generics Introductions

Essential collection frameworks.

---

oracle Java Documentation. | Java 8 std documentation.
↳ docs.oracle.com/javase/8/docs/ (Help file for referance) *

James Gosling, Mike sheridan, Patrick Naughton initiated java proj in 1991
Java originally developed by JG in 1995 released by sunmicro systems.
First public implementation of Java 1.0 released in 1996.
Java is General purpose, concurrent, class based, OOPs language.
Java is statically typed ie, types of variables known at compile time.
Java provides write once, Run anywhere functionality
Java is High level language.

JavaSE8 | JavaSE11 These are LTS long Term support.

All major features java developed in them. SE16 also LTS.

Java SE 08 : March 18, 2014 | JavaSE8 : Sept. 25, 2018 | March 16, 2021.

Java Platforms

- Java SE : It's std. Edition | core Java | General purpose Desktop, etc
- Java EE : It's Ent. Edition | Adv/ent | web java | web services
- Java ME : It's Micro Edition | TV, STBs, printers, etc
- Java Card : A technology that allows small java based applications to be run securely on smart cards & small memory devices. (API : Application Program Interface).

SDK, JDK, JRE, JVM

SDK : software development kit
It's combination of development tools, documentation, libraries, Runtime enviroment. It's like OS without OS one can't write program.

JDK : Java Development kit
It's combination of java development tools, java documentations, rt.jar & JVM. (They gives runtime enviroment)

JRE : Java Runtime Enviroment.
It consists of rt.jar and Java Virtual Machine ie JVM. It's the software comes with JDK. It can be downloaded seperately.

rt.jar file contains core Java API in the compiled form.

JVM : It's an engine which manages execution of java application
It's execution engine.

In java, everything written inside class. There's different types of classes in every different packages.

There are different types of classes available within which codes | implementation is written or done. So there must be .java file so it must be compiled .class file

When JDK installed, there are two folders namely
↳ src ↳ bin

.java files are in src & .class are in library of rt.jar file which contained compiled API files.

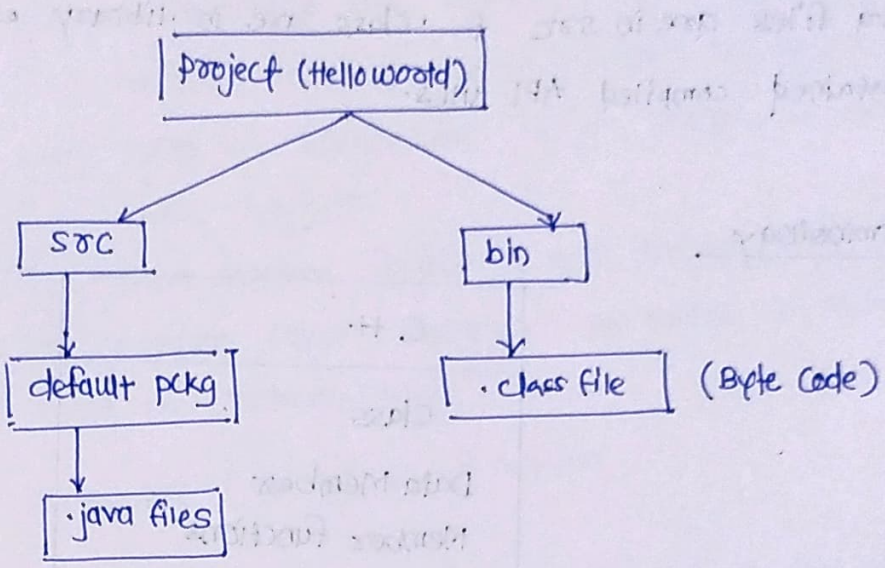
Bhushan Pawar
@Youtuber

Terminology

C++	Java
Class	Class
Data Member	Field
Member functions	Methods
objects	Instances
pointers	References
Access specifiers	Access Modifiers
Base class	Super class
Derived class	Sub class
Derived From	Extended from
Runtime Polymorphism	Dynamic Method Dispatch

package : It's kind of folder only where data is segrigated as per required.

Bhushan Pawar
@Youtuber



Hierarchy for Project

In java also there's entry pt. function to start program in java.

✓ public static void main (String args [ ]) — syntax

In java everything, must be inside class only.

.class files are equals to number of class inside program

For compilation in notepad – javac <filename>.java.

<filename>.java

.class basically referred as Byte code

public static void main (String args[]).

public ↳ It should be accessible outside scope of local class. main is called by operating systems. But operating system is not in program it's outside so main is public.

static ↳ Always, functions are within class and object is call for function / method. And static methods are called with help of classname ie, if static declared, no need to create object.

✓ In above syntax, object is not created still main() is called.

void ↳ No return from main(), no value is returned. void is return only.

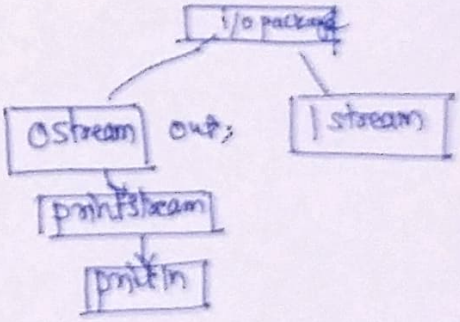
main ↳ Keyword / entry pt. function.

Bhushan Pawar
@Youtuber

Command Line Arguments ↳ (String args[]) : To store into argument array of type string ie (String args[]). String is class which is predefined. (All class name start Capital letter)

↳ println() : It's function print line used to print data on screen. Only println() not work as java strictly follows hierarchy of methods & functions. println() function is defined within PrintStream class. It's (PrintStream) class which is defined inside java.io.printstream

println() is hierarchy like →



java.io.printStream. In io package there's Ostream & Istream

out is object. To call println

object name . function name

so, out.println.

By default there's java.lang pkg, there's system class inside this there's method and ie static method, so to call that object with class name. ⇒ System.out.println();

Program Helloworld

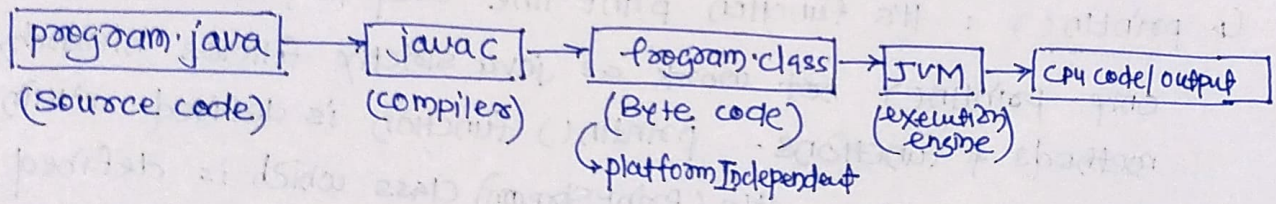
```
class Helloworld {
    public static void main (String args[])
    {
        System.out.println ("Hello world");
    }
}
```

Bhushan Pawar
@Youtuber

Always file name of program and class name in which main() is there should be same so to understand compiler in which class there's main().

If same name given to file name & other classes too, then compiler asks which main class to run. (Conflict) But if class name same as file name & that class doesn't have main() then its compile time error.

Flow of execution of program :-



Function overloading

In same function, within same scope, different no. of parameters is called function overloading.

main function can be overloaded, but provided static function call

```
Class f1 {
    static void main (int number)   // declared static
    {
      system.out.println (number) ;
    }
    public static void main (string [] args)
    {
      system.out.println(" Inside");   → f1 p = new f1 (object created)
      f1.main (25) ;   // object name
    }
```

Bhushan Pawar
@Youtuber

```
O/P = Inside
      25
```

In java, function within class declare it as static, so that call given to function.

Static function can access only static data.

How to create package ?

↳ create new package ↳ com.sunbeam.ob

↳ create java file.

Then java file available only that package.

```
public class f2 {
    public static void main (String [] args)
    {
        int num = 45;
        System.out.println ("Number = " + num);
    }
}
```

↳ O/P = 45

It's good practice to keep all files in own package.

package used to segrigate data, programs.

Bhushan Pawar
@Youtuber

Data Types

Data types given to variables to find type of parameter.

↳ Primitive
It has fix values
(value types)
↳ Byte, char, short
int, long,
float, double.

↳ Non-primitive
It's referance types.
↳ Interface, classes
Type variables, Array.

SrNo	Type	size	default value	Wrapper
01	boolean	not specify	False	Boolean
02	byte	1	0	Byte
03	char	2	\u0000	characters
04	short	2	0	short
05	int	4	0	Integer
06	float	4	0.f	float
07	double	8	0.0d	Double
08	long	8	0L	Long

In java, there's primitive & non primitive data types.
In non-primitive, java giving referance type or non primitive operation. Every primitive type having respective class name as Integer, float, chara, double they can be represented in non primitive type by specific class name which is predefined w.r.t. primitive type. These ref. type of parameter's called as wrapper class.

They (wrapper class) defined in java.lang package.

When value is needed to pass as referance value type then non-primitive type used, as java doesn't support pointers.

All wrapper classes are final ie constants.

Need of wrapper class :

To convert non-primitive to primitive or vise versa wrapper class needed.

eg. If enter ph.No. → Through Command Line Arg. → dataType is string

↳ string is nonprimitive → "12345678510" (string) → Need to convt. to int

↳ wrapper class.

Bhushan Pawar @ Youtuber

Youtuber Bhushan

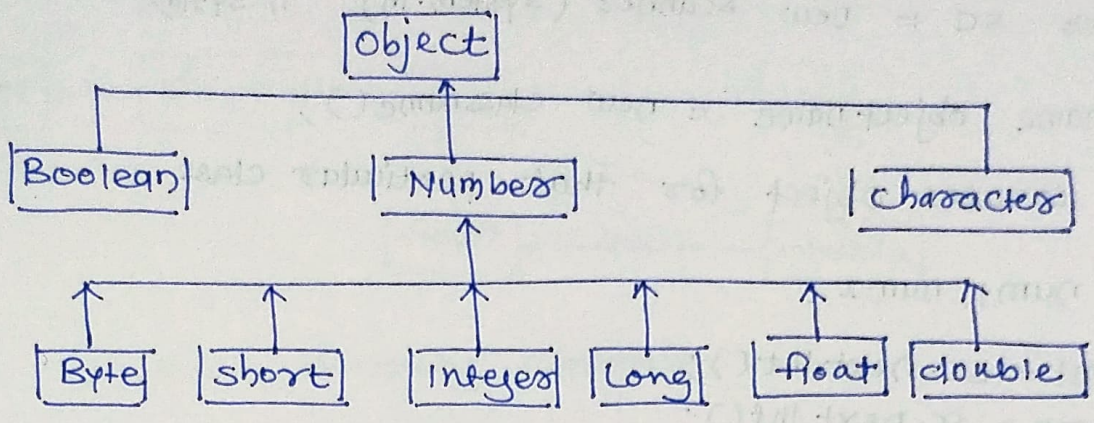
eg. if salary i/p taken "50,000" through CLA then its string. If want to add 6,000 then it's not allowed. so need to convert 50,000 string into int using wrapper class; after that addition is possible.

To store value of primitive type into instances of generic class, type argument must wrapper class

✓ Stack <Integer> stk = new Stack <Integer> ();

× Stack <int> stk = newstack <int> (); //Not allowed

Wrapper class



Object is supermost class in the java.

How to take input from user

There are following methods for taking i/p from user

- console class
- Scanner class
- Command Line Arguments
- Data Input Streams class

Normally, Scanner class used.

- Command line Arguments :

Requirement : Pass 2 CLA | num1 & num2 | num1+num2.

```
public class Q1
{ public static void main (String[] args)
{
    // int num1 = args [0]; } Type mismatch error : can't convert
    // int num2 = args [1]; } from string to Int (need wrapper class)
int num1 = Integer. parseInt (args [0]); // parseint is function used to
int num2 = Integer. parseInt (args [1]); // type is Integer & function is  string to Integer. It's return
                                         // Static. so write Classname. fun()
System.out.println ("R.: " + (num1+num2));
}
}
```

// if argument n't passed Array Index Out of Bound Exception error

---

Run → 1) javac <filename>.java
→ 2) java <filename> <Arguments>

How to take i/p from user : Scanner class.

#✓ import java.util.*;

✓ Scanner sc = new Scanner (System.in) // syntax

// class name object name = new classname();

new creates object for that pertiular class

✓ int num1; num2;

num1 = sc.nextInt();
num2 = sc.nextInt();

✓ System.out.println ("Addition = " + (num1+num2));

# for integer ⇒ sc.nextInt();	for float ⇒ sc.nextFloat();
# double ⇒ sc.nextDouble();	for string ⇒ sc.nextLine();

Typecasting : (Primitive to primitive conversion)

↳ Narrowing ↳ widening

* Conversion of the one data type into another called as Typecasting.
* To convert double (8Byte) to Integer (4Byte) called Narrowing
* To convert Integer (4Byte) to double (8Byte) called widening.

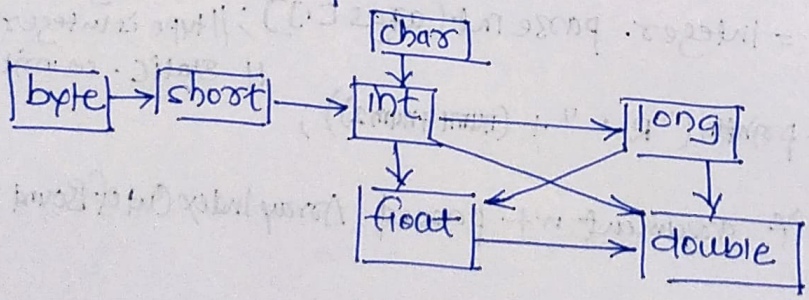
Smaller size to bigger size, due to this no data loss

but in narrowing bigger size datatype converted into smaller size

there's loss of the data.

eg. double d = 50.5;
int num = (int)d; | O/P = 50 (0.5 is lost due to narrowing)

eg. int num = 35;
double d;
d = num; (double)num | widening
O/P = 35.00



— fig — Widening Conversions —

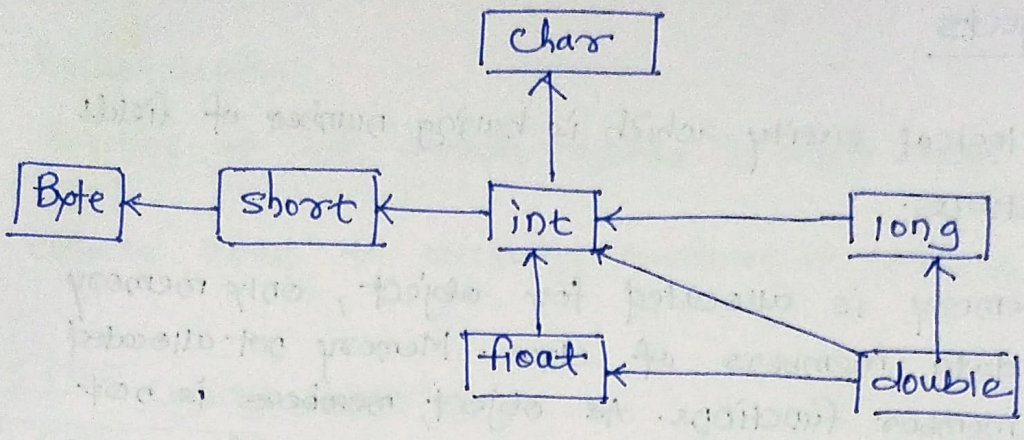


fig — Narrowing conversion —

Boxing :> (Primitive into Non-primitive conversion)
Unboxing :> Non primitive into primitive conversion)
all small case int, char, bool, byte → primitive
All Capital case Int, Char, Bool, Byte → Non primitive type.

```
public static void main (String [] args) {
    String str = "123";
    int number = Integer.parseInt(str);   // unboxing
    System.out.println("Number : " + Number);
}
```

```
public class Q2 {
    public static void main(String [] args) {
        int num = 123;
        String str = Integer.toString(number);
        System.out.println("str = " + str);
        int val = 420;
        String str1 = String.valueOf(val);
        system.out.println("val = " + val);
```

Method-1 Program to convert int to string

method-2

Class And objects

A class is logical entity which is having number of fields & member functions.

Whenever memory is allocated for object, only memory allocated to data members of class. Memory not allocated for object member functions. As object, ~~members~~ is not necessary to perform all member functions of class.

Class
(logical)

employee

id salary name
code() test() write() read()

Fields
Members
elements
Attributes

methods
functions

object
(Physical)

e1 Rahul

id salary name

e2 Bhushem

id salary name

Class Members = fields + functions

e1 Rahul can code(), test(), write(), etc but e2 Bhushan can code() only. It's not necessary both should same.

Local entity

```
Class Person {
    string name;
    int age;
    int id;

    void disp()
    {
      ---
    }
    void read()
    {
      ---
    }
}
```

Scope

Fields : {
name, age, id

Methods : {
disp(), read()

Class binds fields and methods in single class

Class is eg. of encapsulation

Person p1
P1 is object

string name; int age; int id;

disp()

Person P2
P2 is object

string name; int age; int id;

disp()
read()

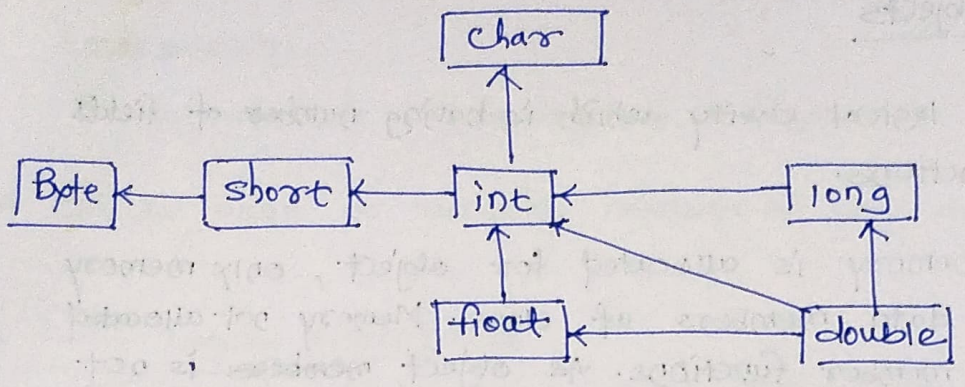


fig — Narrowing conversion —

Boxing :> (primitive into Non-primitive conversion)
Unboxing :> Non primitive into primitive conversion)
all small case int, char, bool, byte → primitive
All Capital case Int, Char, Bool, Byte → Non primitive type.

```
public static void main (string [] args) {
    string str = "123";
    int number = Integer.parseInt (str) ;   // unboxing
    System.out.println ("Number : " + Number);
}
```

```
public class Q2 {
    public static void main (string [] args) {
        int num = 123;
        string str = Integer.toString (number);
        System.out.println ("str = " + str);
```

} Method-1 program to convert int to string

```
        int val = 420;
        string str1 = string.valueOf (val);
        system.out.println ("val = " + val);
```

} method-2

Encapsulation:

Binding of code & data together in single scope ex- class.

Objects used to accessing members of class ie, instance of class.

Class Logical related data

Car
- color - name - brand

Any logically related things placed in one scope programically called class.

Person
- name - id - age - salary

→ calMiles perHr()
↳ dispInfo() } Attributes | datamembers | fields.

→ walk()
↳ read()
↳ code() } Fields

Fruit
- name - color - taste

→ Available()
→ dispInfo() } fields

In java, behavioural things are methods of class
eg. Car has to calculate CalMiles perHr(), dispInfo()

Object is physical entity for which specific properties are there.

ie Person

Bhushan	Rahul
- walk() - talk() - code	- walk() - talk() - read()

Here, Bhushan & Rahul are objects as they have diff. specific properties but generally they are person ie class

Whenever, class is created, no memory is allocated but, when object is created, memory is allocated.

Class = fields (data members) + Methods (member functions)

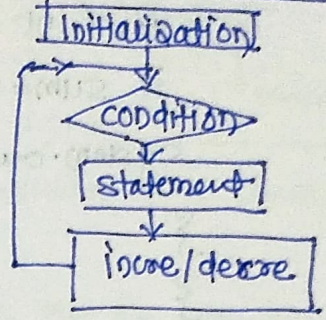
Day 03 : 9/9/2022

✓ Numeber pattern programming | for Loops | While loops | Do while loop

Loops : To repeat entire process or certain part of code "Loop is used".
To stop loop, condition required.

✓ steps : 1 Intitilization (starting pt.)
✓ Step 2 : Condition (loop to stop)
✓ Step 3 : Increment / Decrement.

Initialization is done once only.



Loop structure

✓ # For Loop :

Bhushan Pawar @Youtuber

Syntax

```
for ( initialization ; condition ; Incre/Decre)
```

```
for (int i=0 ; i<=10 ; i++)
```
— forward 0 to 10

```
for ( int i = 10 ; i >= 0 ; i--)
```
— Reverse 10 to 0.

Intialization is done in the loop.
Matrix pattern: for loop used nesting.

✓ # While loop :

```
while (condition)
  {
    // code; i++/i--;
  }
```

if condition is True, then only loop executed.
if condition is false, no execution.
While loop exits once condition is false.

Intialization is to be done before the loop.
For loop used for series type programming.
Every prog. can be implemented with both loops.

In while loop to enter condition is required to be True for execut but in for loop control is in loop.

For loop to enter condition is not needed to be True to execute or to enter inside the loop.

✓ # To increament / decrement by 10/20/x : ; i-10 / i+10 / i±x)
3rd step.

Sum of natural numbers 0 to 25 Numbers - # For loop - H.W. - Whileloop -

```
Class Sumofnaturalnumber {
public static void main (String args[]) {
    int num=25
    int sum=0

    for(int i=1 ; i<=25 ; i++)
    { sum= sum+i }

    System.out.println("sum"+sum) ;
}
}
```

int sum=0 ; initialized once
if initialized once, it will [illegible]
due to re-initialization.

sum of 0-25 natural number.

sum = sum+1 = 0+1=1
Sum = Sum+2 = 1+2=3
sum+3 = 3+3 = 6
sum+4 = 6+4 = 10
loop goes on.

end of program

Break : It will terminate loop immidiately.
Countinue : jump to next iteration.

While loop

```
Class Whileloop {
public static void main (String args[]) {
int i=1 ;
while ( i<=10)
{
    System.out.println (i) ;
    // need terminating cond^n
}
```

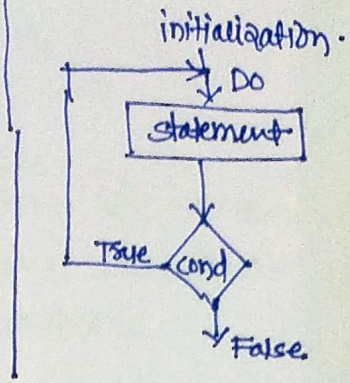
statement for infinite loop
value of i needs to be changed.
To avoid ∞loop add i++ in code.

Do while loop :

```
do {
    // code

} while ( ) ;
```

```
do {
    // code of execution
    // print // i++ ;
} while (condition) ;
```



Whenever do while loop runs, its exit controlled.

Atleast Do-while loop runs for once. It check condition after checking or executing code.

If condition is false, then that time also, loop run min. for one time.

Ex: ATM m/c before checking balance, you need to swipe & popup mes there that's do while loop.

Before checking condition prior use do while loop.

Menu driven programs written in dowhile series,

In this loop, in do { loop } modified value is checked then while (condition) is checked.

To avoid ∞ loop, increment / decrement ~~ther~~ needed.

✓ # Basic code.

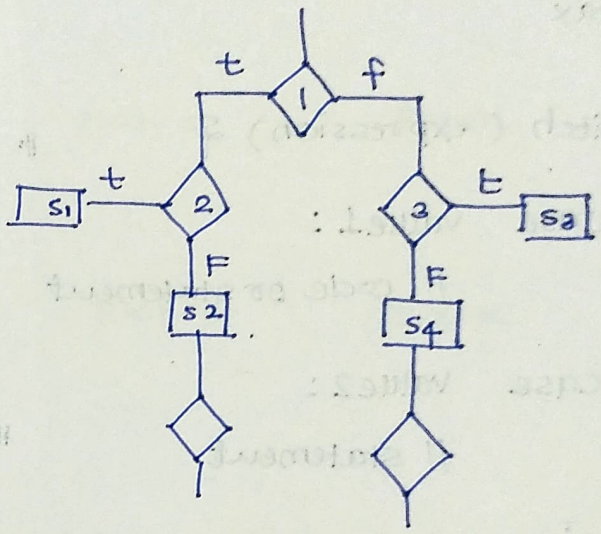
```
Class Dowhileloop {
public static void main (String args[]) {
int i=0;
do
 {
   ++i ; // i++    : no effect on o/p. same o/p printed.
   System.out.println (i) ;
 } while (i<10) ;
```

Day 04 16/09/2022

switch case:

In if there's one condition, in if else - 2 conditions are there. Sometimes, nested if's also used for diff. types of conditions. Sometimes for nesting hierarchy is created for conditions

```
if (cond)
{
    // code; S1
    if(cond)
    {
        // code S2
    }
    else
    {
        // code
    }
}
else
{
    // code
    if (cond)
    {
        // code; S3
    }
    else
    {
        // code; S4
    }
}
```



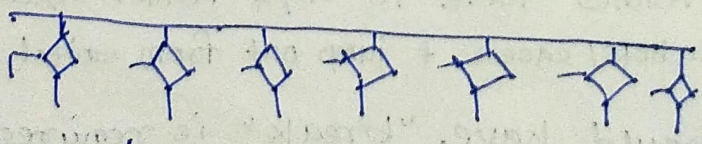
For more conditions and structures becoming more complex.

For different types of conditions if-else loop becomes hierarchy of if-else loop.

Also, its mandatory to check first condition then only it will enter into loop that's restrictions.

To combine more conditions this if-else approch is difficult & complex.

Parallel condition checking is difficult.



(Parallel conditions)

7 days ≠ diff. meetings

For parallel conditions checking switch case is used.

The switch statements is multiway branch statement. It provides easy way to dispatch execution to different parts of code based upon value of expression. As such it often provides a better alternative than complex large series of if-else statement.

Syntax

Bhushan Pawar @Youtuber

```
switch ( expression) {
    case  value1 :
          // code or statement
    case  value2 :
          // statement.
    :
    :
    case  valueN :
          // Statement
        break ;
    default :
        // Statement
}
```

switch (char or byte or int or Num)

Anything like char, byte, int digit, expression, string, enum, etc can be in the expression.

value1, value2, the nomenclature or datatype of it should be same.

case is the keyword

value2 is any datatype, num or anything like identifier too.

Default statement is optional if no case is matched, default statement is executed. If no case is matched & no default case is there, there is action is taken too.

Break statement is used inside the switch to terminate a statement sequence. When break statement is there, when execution reaches there it stops further execution of codes like labels/cases & jump out from switch.

Every case should have "break" is required.

Day 05 : 17/09/2022

Types of variables :

Instance variable | Local variable | static variable.

memory allocated to variable based upon data type used.
It's variable, value can be changed, depending upon that there are types of variables mentioned above.

Basic form

type identifier [= value] [, identifier [= value] ...] ;

Instance variable :-

object ref. is required, to access variable in methods.
Instance variable is variable defined within class.
Each instance contains copy of its own. It can be accessed in other classes too with help of access modifier.

Public	Private	Protected	default.
Anywhere		only child class	within project only.

Local variable.

variables used in methods, constructors, methods, blocks, etc
They are initialized within it only. | Method Name() : method

```
int calculate ()
{
  int i, j ;
  i=10 ; j= 20 ;
}
```

Here, access modifier can't be defined

Static variable -

Static keyword fix the value in it. For variable when static is used then value content can't be changed.

```
Class Employee {
    string name;
    int id;
    static int deptcode=111 ; // static variable.
```

1. Static variable can't be redefined it, only assigned once.
2. final keyword can be added, to make it literal.

   final static int deptcode=111.;
3. static method's can be also defined.

Member function / method

Syntax :

```
<Return type> < method name> < ( parameter list) > ;
{
 // scope of method
}
```

Based on parameter list eg addition, sub, etc mention return type eg. int.

Bhushan Pawar @Youtuber

method declaration : semicolon in method.

eg :

```
< Return type> < method name> < (parameter list) > ;
```

Method defination :

```
<Return type> < method name> < (parameter list)>
{
  //scope of method
}
```

For method name always use "camel case." eg. doubleDiscount. It's part of naming conventions.

When ~~access mo~~ range of method is needed to define then access modifiers used not specifiers.

Day 19/09/2022

In same program one or more program classes can be created. In one class there's method, in another there's call from main function.

```
Class Op.1 {
    static int sum (int i, int j)
    { return i+j ;
    }

Class Op.2 {
    public static void main {
    System.out.println(op.1.sum(11,22)); }
```

In this case, main method class should be compiled.

```
Op.1 op1 = new Operation1();
```

Add foreachloop

For multiple classes, there's multiple classes in a program in one folder called user defined package.

It's good practise, in one class everything inputs, data and call them in another class for security.

Execution of class : driver class

method overloading : Name of method same.

Type 1	No. of parameters.
Type 2	Data Types.

Type1 : No. of parameters :

Compiler compares based on no. of parameters, which method to call according to that output is shown.

Type 2: Data Types.

Compiler compares based on no. of parameters & data type which method to call according to that o/p is shown. values at available at compile time.

H.W - WAP to print and overload method "print" with int | float | double | int, float, double | string

Constructor –

To initialize values before starting program.

Name of const. should be same as class name. It shows behaviour like method but it doesn't have any return type.

Const. has no return type.

Const. empty called as default const. Every prog. should have atleast one default const.

It's purpose to initialize the value. Multiple const. can be written in program; called as constructor overloading with parameters.

```
class student {  int id; string name; int marks;
  student ( ) {  }          — default const.
  student (int x)           — parameterized const.
```

To initialize default values, it can be mentioned in default construction.

Parameterized constructor: passing values will mentioned

```
student (int x)
```

To call constructor, objects needs to be created.

```
student s1 = new student ();
  s1.display ();   // instance call.
```

The moment object created, it will give call to constructor. In execution time when object started executed, constructor initiated. Such programming called as compile time programming.

Never write seperate call for constructor. When values set for compiler at compile time, call compile time.

There's no value in const. called null constructor.

Arrays :

Indexed collection. Array is an object.

Array is indexed collection of similar data element that has fixed size.

Array
↳ define array ↳ create array

Syntax

<datatype> <array name [ ] > ;

or

<data type > [ ] < array name > ;

Bhushan Pawar @Youtuber

Array declaration:

Define array name then allocate memory using 'new' keyword & mention data type. Size of array should be defined in [ ].

<array name> = new < data type> [size] ;

Array declaration & creation can be combined as follows

<data type> <array name> [ ] = new <datatype> [size] ;

eg:- int a[ ] = new int [5]

Purpose of array to create group of similar datatype.

name a	10	20	30	40	50
index →	0	1	2	3	4

values → a[0] = 10 , a[1] = 20 , a[2] = 30 a[3] = 40 a[4] = 50

Index starting from zero Always. Index is used to trace values.

// Calculate size of Array

int a1[] = new int[5]

int = 4 Bytes | size = 5 | Array size, bytes = 4 * 5 = 20 bytes

Array Index of Bound Exception : Index value error | overflowing

// Array defination :

int a1 = {2, 4, 6, 8, 10, 12} ; // Another type of declaration

It's called as Array initialisation.

int a1[] = new int[20] → | size = 20 Bytes |

SOP (a1.lenght) = Gives array lenght = 20 is lenght.

SOP (a1 [2]) = Gives o/p at index value is 3 ~~in this example~~

SOP (a1[10]) = Array Index of Bound Exception error

This is 1-D Array it can be vertical or horizontal.
It has only lenght. It can have only one dim.

Array Declaration

```
int a1[] = new int[5]
int a1[];
int []a1;
int a1[], b1[], c1[];
int a1[], b, c ;
int []a1, b, c ;
```

Array creation

a1 = new int[5]

For CDAC Related
Subscribe channel
@Youtuber
Bhushan Pawar