Public class will only be the class that has name of the file

File name: Demo.java Class name: Demo

It's a good practice to use the initial letter as capital (not necessary).

public: This keyword means we can access the class from anywhere

functions: Collection of code (Group of statements) that we can use again & again. Also known as "Methods"

void: The void keyword specifies that a method should not have a return value

string[] args: Means an array of sequence of characters ("strings") that are passed to the main function

array

- After compiling, .class file is always saved in current location where you are in.
- If you want to change the location, use -d (destination) option while compiling & specify the path.

. = curr. directory .. = prev. directory

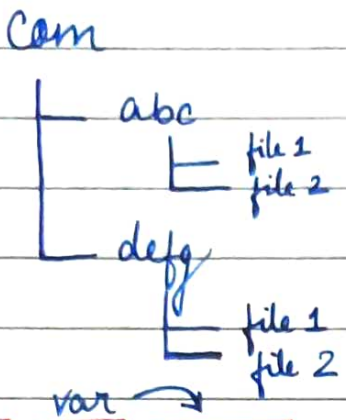
javac -d <path> Demo.java

- echo $PATH: Every command looks for this location before executing

[environment variables]

- Class name and file name should be same, but if we don't want to make class name as file name then it should not be public.

package com.abc OR package com.defg



System.out.println("Hello"); → This means print the output on Standard output stream (here, terminal)

System → class
out → var
println → fn / method
out → Standard o/p stream

println → adds new line

print → does not add new line.

- Scanner input = new Scanner(System.in);
  - Scanner → class that allows us to take input (available in java.util package)
  - input → user-defined
  - new → keyword, creating objects
  - Scanner(System.in) → constructor to initialize; take input from standard input (here, keyboard). Standard i/p stream

Primitive → means any datatype that cannot be broken further.
for eg: integer, character, etc.

⇒
```
int    rollNo = 64;                          → 4 bytes
char   letter = 'r';
float  marks = 98.67f;                       → 4 bytes
double largeDecimalNumbers = 456789.12345;   → 8 bytes
long   largeIntegers = 12345678910L;         → 8 bytes
boolean check = true;
```

- String is written in double quotes whereas while specifying char we write it in single quotes.

float marks = 7.2(f)

(by default) double large DecimalNumbers = 456789101.12345

(by default) int rollno = 64;

long largeInteger = 12345678910(L);

- Integer → WRAPPER CLASS → provides additional functionalities
  → converts primitive datatypes to objects

- Comment → The lines that we comment are ignored by Java & will not be executed.
  Denoted by //

int a = 10
identifier ← a, 10 → literal

- Literals : Java literals are syntax representations of boolean, character, numeric or string data.
  Here, 10 is an integer literal.

- Identifiers : Identifiers are the names of variables, methods, classes, packages and interfaces.

- int a = 234_000_000;
  → the value of 'a' will be 234000000, underscore will be ignored.

- 564.12345678 —round off→ 564.12345
  If we give float very big, than it rounds off the value which gives floating point error.

TYPE CASTING & TYPE CONVERSION

* Widening OR Automatic Type Conversion :
  - Two datatypes are automatically converted.
  - This happens when we assign value of smaller datatype to be compatible with bigger datatype & two datatype must.

byte → short → int → long → float → double

Eg: int i = 100; → 100
long l = i; → 100
float f = l; → 100.0

* Narrowing or Explicit Conversion:
- This happens when we want to assign a value of larger data type to a smaller data type.
- We perform explicit type casting or narrowing.

double → float → long → int → short → byte

Eg: double d = 100.04; → 100.04
long l = (long) d; → 100
int i = (int) l; → 100

* Automatic Type Promotion in Expressions:
- While evaluating expressions, the intermediate value may exceed the range of operands and hence the expression value will be promoted.
- Some conditions of type promotion are:
  1) Java automatically promotes each byte, short, char to int when evaluating an expression.
  2) long, float or double the whole expression is promoted to long, float or double.

Eg: After solving expression:
(f * b) + (i / c) - (d * s);
we get → float + int - double = double
(float: converted to biggest one)

* Explicit Type Casting in Expressions:

- If we want to store large value into small data type.

Eg. byte b = 50;

b = (byte)(b*2); ⟶ type casting int to byte.