First Java Program - Input, Output, Debugging & Data types

* File name Demo.java

↓

Class name - Demo

↳ good practice to make initial capital

* public - means can be used and called from anywhere

* functions - also known as methods

↳ Block of code

* void - methoud should not have return type

* String[] args - Command line arguements

↳ [] array → array of sequence of characters passed to main functions

* .class file always saved in same location

↓

to change -d flag is used

↓

e.g. javac -d <path> Demo.java

* package - com.atharva OR com.abc

```
com
 ├ abc ─┬ file 1
 │      └ file 2
 └ atharva ─┬ file 1
            └ file 2
```

var

* System . out . println (" Hello ");

↳ class

funct^n / method

this means that print the output on standard output stream

↓

here terminal

* println - adds new line

* print - does not add new ~~f~~ line

* Scanner input = new Scanner (System.in)

Class that allows us to take input

creating object

take i/p from standard input (here keyboard)

* Primitive - means any data type that cannot be broke further

⇒ int rno= 64; → 4 bytes

char let = 'a';

float m= 98.67f; → 4 bytes

double n= 45678.12345 → 8 bytes

long n= 4567812345L → 8 bytes

boolean res = true;

* String → double quotes " "

* Char → single quotes ' '

* float → f at end → 98.6f
* long → L at end → 12345L
* default → decimal → float ✗
  ↳ double ✓
  → int ✓
  → long ✗

* Integer wrapper class – provide additional features
  ↳ create primitive data type to object

* Comment in java – ignored lines of code
  ↳ will not execute
  ↳ start with //

* int a = 10 → literal
  ↳ identifier

* Litrals – syntactic representation of boolean, char, int, or string data
  → here 10 is integer literal

* Identifiers – They are the names of variables, methods, class, package, interface.

* Type casting and Type conversion

* Widening or Automatic Type conversion

↳ Smaller data type to bigger datatype

→ byte → short → int → long → float → double

* Narrowing / Explicit conversion

↳ larger data type to smaller data type

→ double, float, long, int, short, byte

e.g.

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

* Automatic Type promotion in expression-

→ While evaluating expressions, the intermediate value may exceed the range of operands & hence the expression value will be promoted.

→ Some conditions-

1) Java automatically promotes each byte, short, char to int when evaluating expression.

2) Long, float / double the whole expression is promoted to long, float or double.

e.g. after solving expression

(f*b) + (i/c) - (d*s)

we get

$$\underbrace{\text{float + int - double}}_{\text{converted to biggest one}} = \text{double}$$

* Explicit type casting in expressions:

→ If we want to store large value into smaller data types

e.g.

```
    byte b= 50;
    b= (byte) (b*2) → type casting int to
                      byte
```

* If... else syntax in Java

```
  if (condition) {
         // block of code
  } else {
         // block of code
  }
```

* for loop syntax

```
 for (statement 1; statement2; statement3) {
              // code block
 }
```