Hai Demo.java

↓

This file should contain a class and name of that class should be Demo.

Because Demo is the name of the file.

class Demo{

}

↳ By convention, the variable starting with capital letter is a class.

When the name of the file is Demo.java, then the Demo class is going to be a public class which means this class can be accessed from anywhere i.e other files, packages etc.

Date:

Main ():

In Java, program starts from main function. If there is no main function, we will not be able to run the program.

To compile a .java file

```
javac Demo.java
```

To run

```
java Main Demo
```

(Main → name of the file)

```
public class Demo {
```

public: This class can be accessed from anywhere.

class: named group of properties & functions. Basically all the functions that are inside the class are methods

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

public: Since main() is an important part of program without which code cannot be run, It's a convention to be public so that it can be accessed from anywhere.
If we made it as private, then main() will not be accessible outside because of which the class cannot run.
This is the reason behind main() as public

Date: YOUVA

Static:

1. main() is a part of Demo class
2. program is going to start from this main() method.

Basically in classes, we will be having variables and functions. In order to use them, we need to create the object of the class.

```
public class Demo {
    Demo obj = new Demo();
    obj.main();
    public static void main(String[] args) {
        System.out.println("Hello");
    }
}
```

creation of object — this code will not run

command lined args

Here main() is a method inside Demo class and if we want to acces use that method, we need to creat an object of that Demo class

But the program execution starts from main() so the code above it will not run.

Hence static keyword is used so that we can use the main() without creting the Object of the class.

Void:

Basically when a function stops finishes execution it returns a value. Such as int, name etc.

Here the function is not returning anything hence void.

```
public static void main (String[] args) {
    System.out.println(args[0]);
}
```

↓ prints the first element of array

```
javac Demo.java
java Demo 30
30
```

↳ whatever the command given in the terminal, it is stored as an array in args variable

```
System.out.println(args[1]);
javac Demo.java
java Demo 30 "Java"
Java
```

To change the location of a byte code

```
javac -d .. Demo.java
```

↙ asks for location ↘ previous directory to the Demo.java file's directory

Location of 'javac' and 'java':

The commands that we use javac and java are executable files located in the computer

where javac

/usr/bin/javac

ls /usr/bin | grep javac

open /usr/bin

The computer opens these files via path and environment variables.

package:

The folder in which java file lies.
To allow the accessibility of the file to other files in a particular package.

Outputs:

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

System — a class which contains variables such as out and Methods as println
out — type PrintStream
println — takes a string and outputs it

Inputs:

```
import java.util.Scanner;

Scanner input = new Scanner(System.in);
System.out.println(input.nextL());
```

Primitive Data Types:

DT which cannot be broken down further.
Ex: int
String can be broken down to char

```
int rollno = 64;
char letter = 'r';
float marks = 98.67f;
double largeDecimalNumbers = 456789.5342;
long longInteger = 34765321691L;
boolean check = false;
```

Int Input program:

```
public class Input{
    public static void main(String[] args){
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter some input:");
        int rollno = input.nextInt();
        System.out.println("your rollno is: " + rollno);
    }
}
```

Literals and Identifiers:

```
int a = 10;
```

↙ Identifiers (reference variable) — name of the variable, packages, class . . .

↳ Literals — Syntactical representation of boolean char, numeric . . .

```
int a = 234_000_000
```

to write in terms of mn, we underscore

String input:

```
String name = input.next();      → takes first word
String full_name = input.nextLine(); → takes entire string
System.out.println(name);
```

float input:

```
float marks = input.nextFloat();
System.out.println(marks);
```

Date:

Sum of two numbers:

```
public class Sum {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int num1 = input.nextInt();
        int num2 = input.nextInt();
        int sum = num1 + num2;
        System.out.println("Sum is : " + sum);
    }
}
```

Type conversion:

When one type of data is aligned to a another type of variable, an automatic type conversion takes place when following conditions are met.

1. the two types should be compatible.
2. Destination type should be greater than the source type.

Eg:

```
Scanner input = new Scanner(System.in);
float num = input.nextFloat();
System.out.println(num);
```

68

68.0

```
int num = input.nextFloat();
```
→ Gives error

Type casting:

Compressing the bigger type in a smaller type explicitly.

```
int num = (int) (67.56f);
System.out.println(num);
```

67

Automatic Type promotion in expressions:

```
byte a = 50;
byte b = 40;
byte c = 100;
int d = (a*b)/c;
```

The range of byte exceeds in expression (a*b) so an automatic type promotion takes place to int.

```
byte a = 257;
System.out.println(a);
```
↙
outputs 1

→ 257 % 256 = 1

```
int number = 'A';
System.out.println(number);
```
↙
65

Here also from char to int, conversion takes place

Note: Java follows unicode, we can print other lang also.

```
System.out.println("నమస్తే");
```
↙
నమస్తే

Rules for Type promotion:

1. All byte, short and char values are promoted to integers.
2. If any of the operands in multiplication is long, whole operation is promoted to long.

float → float

int * float → float

Ex:

M T W T F S S

Page No.: YOUVA

Date:

```
byte b = 42;
char c = 'a';
short s = 1024;
int i = 1000;
float f = 5.67f;
double d = 0.1234;
double result = (f * b) + (i / c) - (d * s);
System.out.println((f*b) + " " + (i/c) + " " + (d*s));
System.out.println(result);
```

float + int - double = double