I/O and Data types

> Every file that ends with .java is a CLASS

```
import java.util.Scanner;
//Classes are named with first letter capitalised

//This class can be accessed from anywhere
public class Main{
    //program starts -- main fxn (collection of codes that ca

    //methods -- Fxns in classes

    //static -- helps to run the class w/o creating an obj of
    // example : Aish --> obj of class Human

    //void -- return type
    public static void Main(String[] args){
        System.out.println("Hello"); //std output stream
        System.out.print("Hello");
        // System -- class
        // out -- is a reference string of printStream
        Scanner input=new Scanner(System.in); //std input str
        Scanner input=new Scanner(File); //to read input from
    }
}
```

> Package : Folder where the java file exist

PRIMITIVE DATA TYPES : Any datatype that could not be further broken down

- integer(4) | int rollnum=64;
- Character | char letter='r';
- Float(4) | float marks=98.67f;

> All the decimal values are by default Double

- Double(8) (Large Decimal Numbers) | double largeDecimalNum =4567287.5788;
- Long(8) (Large Integer Values) | long largeInt=5127868972391380138913328L;
- Boolean(1 bit) (true / false) | boolean check='false';

NON-PRIMITIVE DATA TYPES :

- String

TYPE CONVERSION /CASTING:

when one type of data is assigned to another type of variable.

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

```
int num=int(657.97f);
System.out.print(num):
```

Automatic Type Promotion in Expressions:

```
int a=257;
```

```
byte b=byte(a)
```

output will be ‘1’

Now , it the remainder of 257 and 256 (257%256=1)

```
byte a=40;
byte b=50;
byte c=100;

int d=(a*b)/c;
System.out.print(d);
```

output will be ‘20’

HOW is (a*b) that is (40*50)=2000 being able to store in byte when the limit is only 256?

Java automatically promotes the integer type while performing the expression (ONLY IN EXPRESSIONS)

- All the byte,Short,Char — integer
- Any operands — Long

```
int num='a';
System.out.print(num);
```

output will be the ASCII value of the alphabet (Unicode)