Introduction to Programming language.

- Programming is a way to instruct the computer to perform various task.
- Computers only understands Binary i.e 0's and 1's
- Instructing computers in Binary i.e 0's and 1's are very difficult for humans so, to solve this issue we have programming languages.

- Programming language :- It is a computer language used by programmers to communicate with computers.

* Types of programming languages.
1. Procedural
2. Functional
3. Object-oriented

- Procedural
  - Specifies a series of well-structured steps and procedures to compose a program.
  - Contains a systematic order of statements functions and commands to complete a task

- Functional
  - Writing a program only in pure functions i.e, never modify variables but only create new ones as an output.
  - Used in a situation where we have to perform lots of different operations on the same set of data like ML.

- Object Oriented.
- Revolves around objects
- Code + Data = objects
- Developed to make it easier to develop, debug reuse and maintain software.

"One programming language can be of all 3 types like - Python."

Java follows procedural and object oriented both types.

* Static VS Dynamic languages.
- Static
  - Perform type checking at compile time
  - Errors will show at compile time
  - Declare datatypes before use.
  - More control.

- Dynamic
  - Perform type checking at runtime.
  - Error might not show till programs run.
  - No need to declare data type of variables.
  - Saves time in writing code but might give error at runtime

* Memory Management
There are 2 types of memory Stack and Heap

when we declare a variable then the reference variable stored in stack memory points to the object of that variable stored in heap memory.
For ex:- a = 10

Here "a" is called reference variable, and "10" is the object of That reference variable

- Reference variable are stored in stack memory.
- Heap memory stores the objects of reference variable.

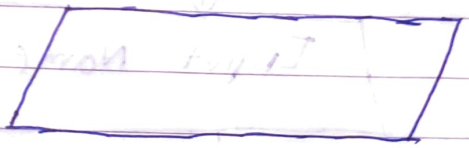
* Lecture -2

- Flow of the program.
- Flow chart :- Visulization of our thought process or Algorithm and represent them diagrammatically is called flow chart.

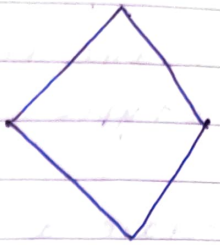
- Symbols to be used in Flow chart

1. Start / Stop —>

2. Input / Output —>



3. Processing —>

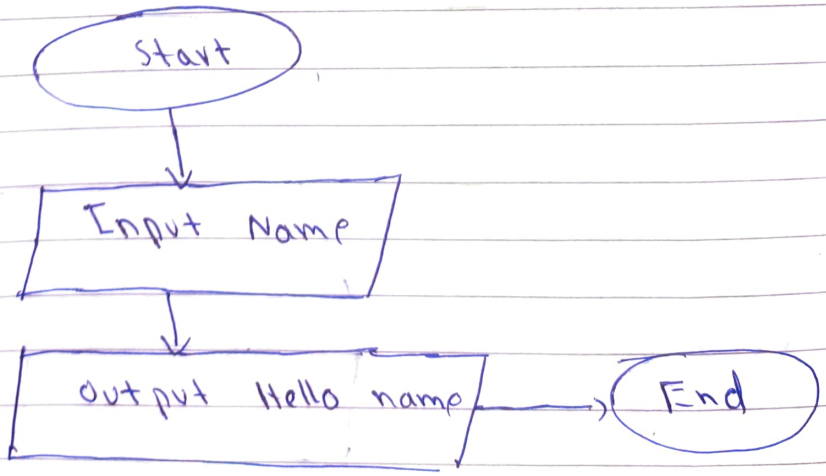
4. Condition ⟶ 

5 Flow direction of program ⟶

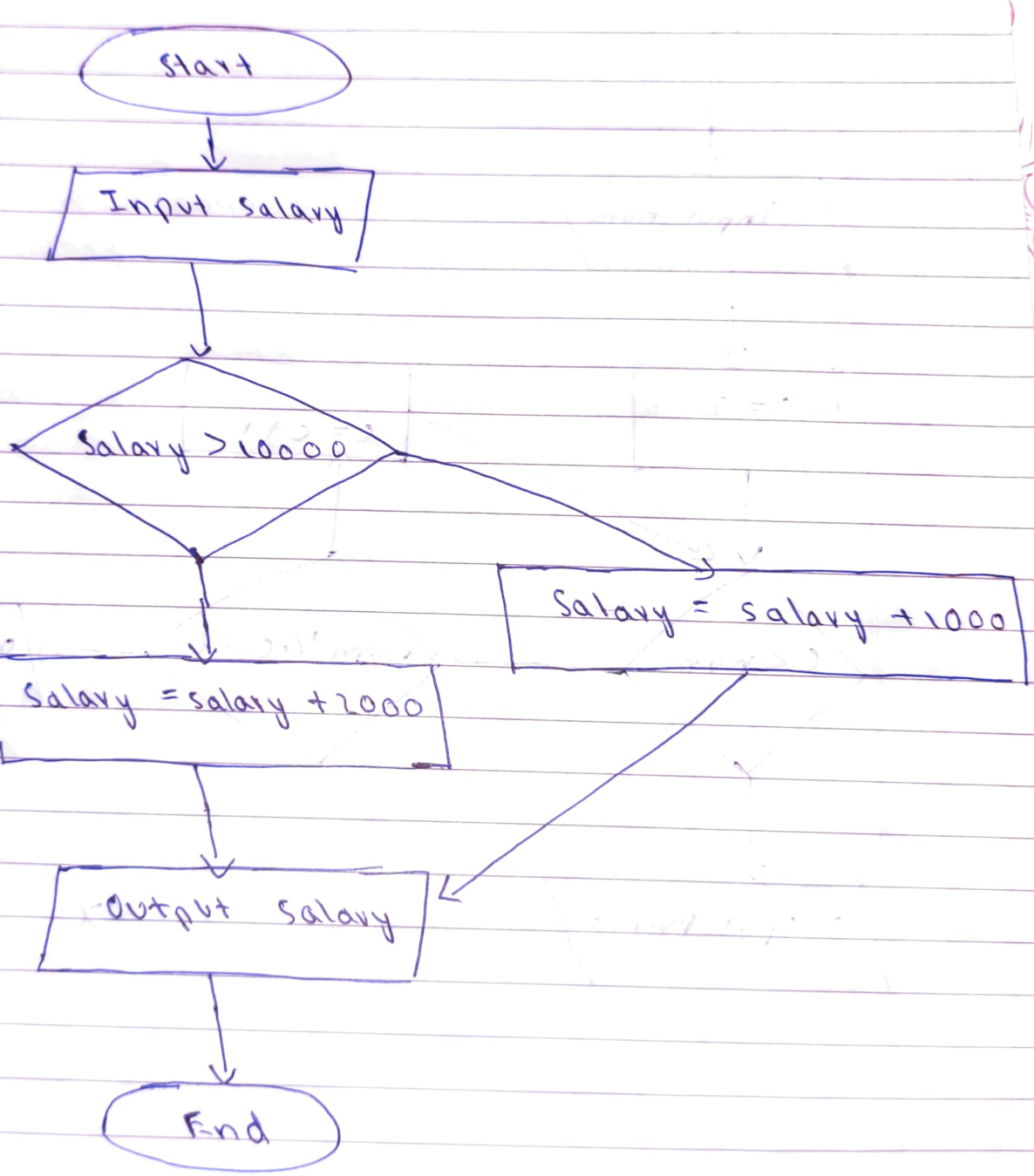
⟶

- start/stop :- An ovel shape indicate the starting and ending points of the flow chart.
- Input/Output :- An parallelogram is used to represent Input and Output in flow chart.
- Processing :- A rectangle is used to represent process such as mathematical computation or variable assignment.
- Condition :- A diamond shape is used to represent conditional statement which results in true or false (Yes or No).
- Flow direction of program :- An arrow shape is used to represent flow of the program.

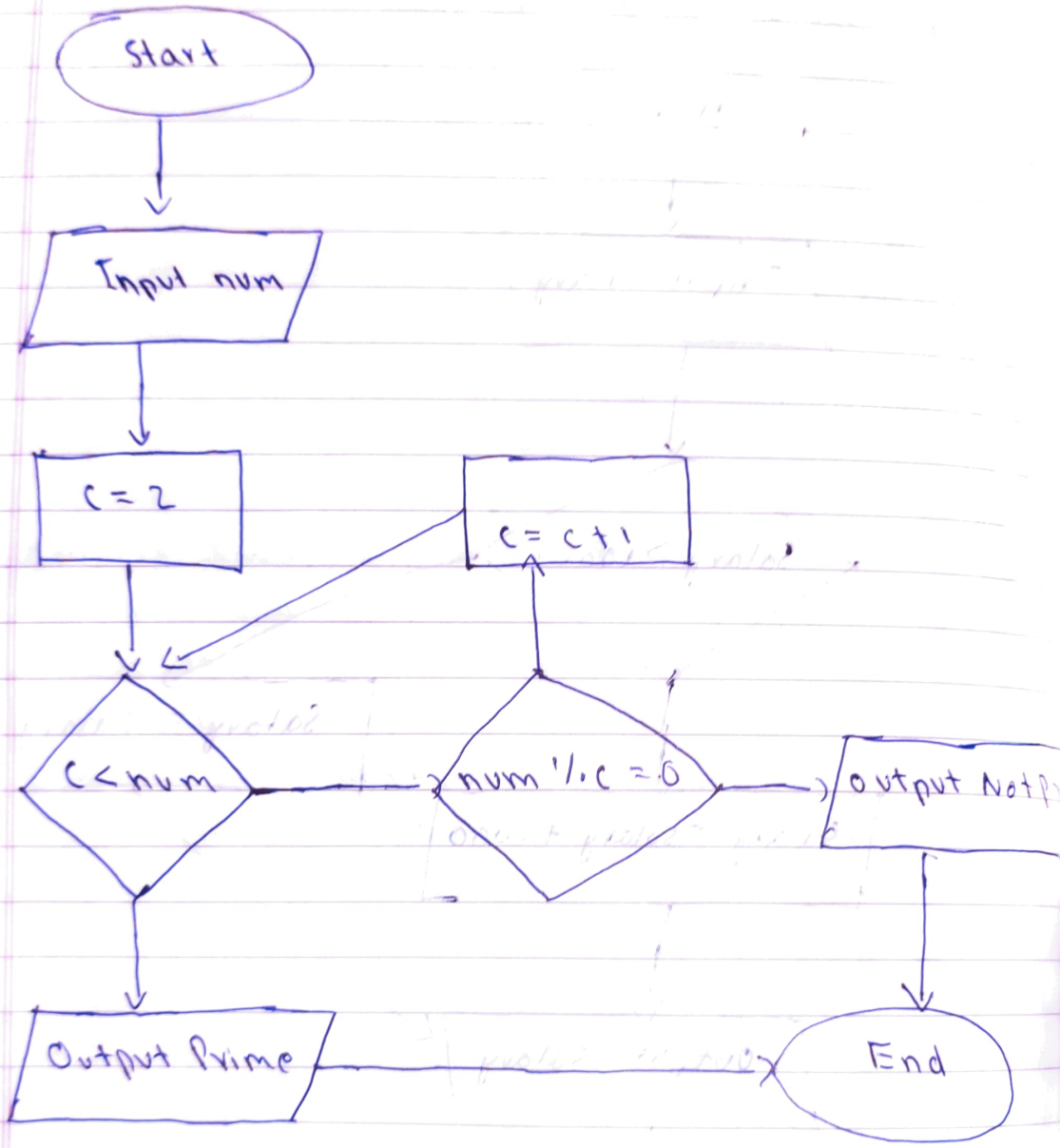
9. Example 1 :- Take a name and output Hello name.



Example 2:- Take input of a salary. If the salary is greater than 10,000 add bonus 2000, otherwise add bonus as 1000.



Q Example 3- Input a number and print whether it is prime or not.



- Pseudocode :- It is like a rough code which represents how the algorithm of a program works.
  - Pseudocode does not require syntax.

- Pseudocode of Example 2

```
start
Input Salary
if Salary > 10000:
    Salary = Salary +2000
else:
    Salary = Salary +1000
Output Salary
exit
```

- Pseudocode of Example 3

```
Start
Input num
if num ≤ 1
    print "Nither prime nor composite"
c = 2
while c < num
  if num % c = 0
      Output "Not Prime"
      Exit
  c = c+1
end while
Output "Prime"
Exit.
```

* Optimization of prime solution

Let's have a number to check it's a prime number of not

-> 36

$$\left[\begin{array}{l} 1 \times 36 = 36 \\ 2 \times 18 = 36 \\ 3 \times 12 = 36 \\ 4 \times 9 = 36 \end{array}\right] \quad \text{.....(i)}$$

$$\left[\begin{array}{l} 6 \times 6 = 36 \\ 9 \times 4 = 36 \\ 12 \times 3 = 36 \\ 18 \times 2 = 36 \\ 36 \times 1 = 36 \end{array}\right] \quad \text{.....(ii)}$$

In the above demonstration we have clearly seen that (i) and (ii) are repeated so, to optimize this we can ignore the (ii)

As same as this

we can check the number is prime or not by travelling from 2 to $\sqrt{\text{number}}$

For example :-

- To check 2345678643 is prime or not, we only have to travel from 2 to $\sqrt{2345678643}$ (i.e. 153156)
- To check 17 is prime or not, we do not have to travel from 2 to 17 we just have to travel from 2 to $\sqrt{17}$ (i.e 4)

* Optimized Pseudocode of Example 3

```
start
input n
if n <= 1 ;
        print(" neither prime nor composite)
    c = 2
    while c*c <= n ;
        if n % c == 0 ;
            Output " not prime"
            ext
        c += 1
    end while
    Output " prime"
exit
```

Lecture - 3

* Introduction to Java

Q Why do we use programming language?

Ans Machine only understand 0's and 1's, for humans it is very difficult to instruct computer in 0's and 1's so to avoid this issue we write our code in human readable language (Programming language)

" Java is one of the programming language"

.java file (Human readable) —compiler→ (Entire file) .class file (Byte code) —Interpreter→ Machine code (0's and 1's) line by line

- The code written in java is human readable and it is saved using extension .java.
- This code is known as source code.

Java Compiler :-

- Java compiler converts the source code into byte code which have the extension .class
- This byte code not directly run on system
- We need JVM to run this
- Reason why java is platform independent

Java interpreter :-

- Convert byte code to machine code i.e 0's and 1's
- It translate the byte code line by line to machine code

* More about platform independent :-

→ It means that byte code can run on all operating system.

→ We need to convert source code to machine code so computer can understand it.

→ Compiler helps in doing this by turning it into executable code

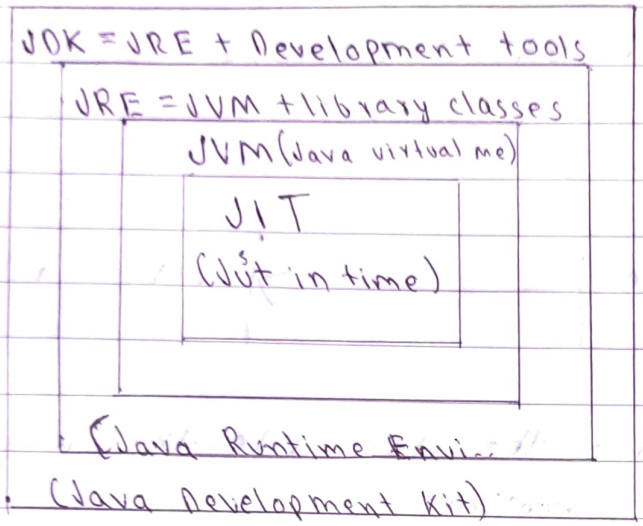
→ This executable code is a set of instruction for the computer.

→ After compiling C/C++ code we get .exe File which is platform dependent.

→ In java we get byte code, JVM converts this to machine code.

→ Java is platform independent but JVM is platform dependent.

* Architecture of Java



JDK

- Provide environment to develop and run the java program.
- It is a package that includes :-

1) Development tools :- To provide an environment to run your program.
2) JRE :- To execute your program.
3) A compiler :- javac
4) Archiver :- jar
5) Docs generator :- Javadoc
6) Interpreter / loader

JRE

- It is an installation package that provides environment to only run the program.
- It consist of :-

1 Development technology
2 User interface toolkit
3 Integration libraries
4 Base libraries
5 JVM :- Java virtual memory

* Compile time :-

- After we get the .class file the next thing happen at runtime,
1. Class loader loads all classes needed to execute the program.
2. JVM sends code to bytecode verifier to check the format of code.

* Runtime :-

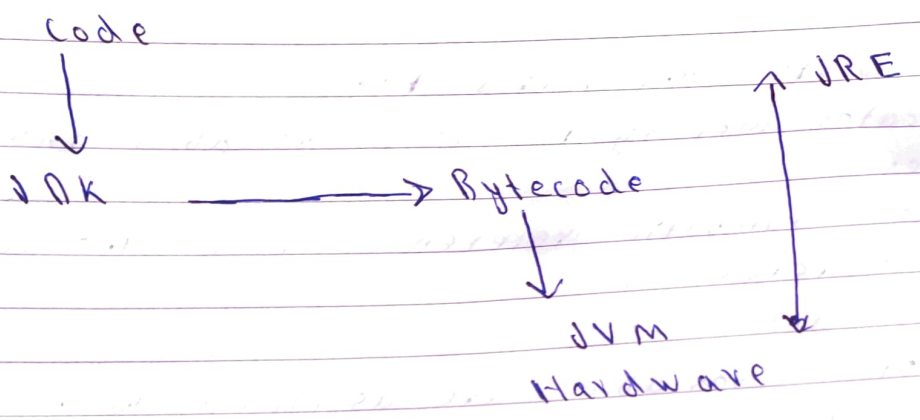
Class loader
↓
Bytecode verifier
↓
Interpreter
↓
Runtime
↓
Hardware

(How JVM works) class loader

- Loading
  -> Read .class file and generate binary data
  -> an object of this class is created in heap.
- linking
  -> JVM verfies the .class file
  -> allocates memory for class variables and default values.
  -> replace symbolic references from the type with direct reference.
- Initialization
  -> All static variables are assigned with their values defined in the code and static ~~bock~~ block
  -> JVM contains the stack and heap memory locations.

* JVM Execution
- Interpreter
  -> line by line execution
  -> When one method is called many times it will interpret again and again.
- JIT
  -> Those methods that are repeated, JIT provides direct machine code so that interpretation is not required.
  -> Make execution Faster
  -> Garbage collector.

* Working of Java Architecture



* Lecture - 4

- First Java program
- Structure of Java File

"Source code that we write will be saved using extension .java".

- Every thing written in .java file must be in classes or we can say that every file having .java extension is a class.
- A class with same name as file name must be present in .java file.

First alphabet of class name can be in upper case. It is the naming convention of class name. however, it is not compulsory to do so

- Class which is having same name as file must be public class.
- A main function/method must be present in the public class, main is a function where the program starts.

- Converting .java to .class
- Using javac compiler we can convert .java file to class

Command to convert .java to .class

Javac and .java file name

let the name of .java file is main, so the command to convert .java to .class is

Javac Main.java

- Above command create a .class file (Main.class) which contains Bytecode.

.java file ——————→ .class file

- Running the program

By using java and name of file we can run the program

- Command > java Main
- Hello world program

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

1. public (in first line) :- public is an access modifier which allows to access the class from anywhere.
2. class :- It is a name group of properties and Functions.
3. Main :- It is just the name of class as same as the name of file.
4. public (in second line) :- It is used to allow the program to use main function from anywhere

5. Static :- It is a keyword which helps the main method to run without using objects.
6. Void :- It is a keyword used when we do not want to return anything from a method / functi
7. main :- It is the name of method.
8. String[] args :- It is a commands line argument of string type array.
9. System :- It is a final class defined in java.lang package.
10. Out :- It is a variable of printstream type which is public and static member field of the system class.
11. println :- It is a method of Printstream class, it prints the arguments passed to it and adds a new line. print can also be used here but it prints only arguments passed to it, it do not adds a new line.

- What is package?

-> It is just a folder in which java files lies
-> It is used to provide some rules and stuff to our programs.

* Primitive data types
- Primitives data types are those data types which is not breakable.

Ex:-
string is not a primitive data type so we can break this data type into char
i.e, string "Kunal" can be divided into
'k' 'u' 'n' 'a' 'l'

But primitives data type are not breakable.
We cannot break a char, int etc.

- list of primitive data types in java are :-

1. int :- int is used to store numeric digits
   example int i = 26;
2. char :- char is used to store character
   example char c = 'A';
3. float :- float is used to store floating point numbers
   example float f = 98.67f;
4. double :- double is used to store larger decimal number
   example double d = 45676.58975;
5. long :- long is used to store numeric digits which is not able to stored in int
   example long l = 158769580691O1;
6. boolean :- It only stores t values i.e, true or false.
   example boolean b = false;

In float and long we have used f and l, it denotes that the number in the variable is float or long type, if we do not use this java consider float valve as double and long valve as int.

- literals :- It is a synthetic representation of boolean, character, string, and numeric data.
  Ex :- int a = 10;
  Here 10 is called literal.

- Identifiers :- name of variable, method, class, packages, etc are known as identifiers.

  Ex :- int a = 10;

  Here ~~to~~ ~~is~~ ~~called~~ ~~literal~~.

  a is identifier.

* Comments in Java

  Comments are something which is written in source code but ignored by compiler.

- Two types of Comment

1. Single line comment :- used to comment down a single line (// is used for it)
2. Multi line comment :- used to comment down multiple lines

   (/* */ used for it)

* Inputs in Java

  We have Scanner class available in java.util package to take input

  To use this class we have to

1. Import java.util package in our file
2. Create object of the scanner class
3. Use that object to take input from the keyboard.

Syntax :-

```
import java.util.Scanner;
public class Main {
    public static void main (String [] args){
        Scanner input = new Scanner (System.in);
    }
}
```

1. Scanner :- It is a class required to take input, it is present in java.util package.
2. Input :- It is an object that we are creating to take input.
3. new :- It is a keyword used to create an object in java.
4. System.in :- System is a class and in is a variable that denotes we are taking input from standard input stream (i.e. keyboard).

- Int input :- nextInt() is a function used to take input of int.

  Syntax :-

```
Scanner input = new Scanner(System.in);
    int rollno = input.nextInt()
```

- float input :- nextFnt() is a function used to take input of int

  Syntax :-

```
Scanner input = new Scanner(System.in);
    float marks = input.nextFloat();
```

- String Input :- Two ways to take string input

1. Using next() method :- It will take word input till a space occurs

   Syntax :-

```
Scanner input = new scanner(system.in);
    String S1 = input.next();
```

   Input :- Hey Knal

   Output :- Hey

2. Using nextline() Method :- It will take an string input including space.

Syntax :-

```
Scanner input = new Scanner (System.in);
        string s2 = input.nextline ();
```

- Sum of two numbers

```
Impovt java.util.scanner;

public class Sum{
    public static void main(String[] args) {
        Scanner input = new scanner (System.in);
        System.out.print("Enter first number");
        int num1 = input.nextInt();
        System.out.print("Enter second number");
        int num2 = input.nextInt();
        int sum = num1 + num2
        System.out.println("Sum = "+sum);

    }
}
```

Output

```
Enter first number 70
Enter second number 80
Sum = 150
```

* Type conversion

When one type of data is assigned to another type of variable an automatic type conversion will take place under some condition.

Conditions :-

1. Two type should be compatible
2. Destination type should be greater then the source type.

* Type Casting

When we convert one type of data to another type is known as type casting

Ex :- int num = (int) (67.504f)

* 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.

There are some condition for type promotion :-

1. Java automatically promotes each byte, short or char operand to int when evaluating an expression.
2. If one operand is a long, float or double the whole expression is promoted to long, float or double respectively

Ex :-

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

Here when a*b occured it become 2000 which is out of the range of byte so here byte is automati-

-ally promoted to int type.

* Example for thorough review concept.

```
Public class TypePromotion {
    public static void main(String [] args){
        byte b = 42;
        char c = 'a';
        short s = 1024;
        int i = 50000;
        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);

    }
}
```

Output

238.14 515 426.3616

626.7784146484375

* Prime number program

```
import java.util.Scanner;
public class Prime {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("Enter a number");
        int n = in.nextInt();
        if (n <=1){
            system.out.println("Neither prime Nor Comp
```

```
    } return ;

    int c = 2 ;
    if (n == 4) {
        System.out.println("Not Prime");
        return ;
    }
    else {
        while (c * c < n) {
            if (n % c == 0) {
                System.out.println("Not Prime");
                return ;
            }
            c = c + 1 ;
        }
        if (c * c > n) {
            system.out.println("Prime");
        }
    }
}
```

- Output :-

1 Please enter a number
1)
Prime

2 Please enter a number
1
Nither prime nor composite

3 Please enter a number
6
Not Prime.

* Example of if statement.
statement inside if statement only executes when condition given in if is true.

```
public class ifstatement {
    public static void main(String[] args) {
        int a = 10;
        if (a == 10) {
            System.out.println("Hello");
```

Output
Hello

* Example of while loop.
statement in while loop run till condition in while loop become false.

```
Public class whileloop {
    public static void main(String[] args) {
        int count = 1;
        while (count != 5) {
        System.out.println("count");
        count ++;
            }
        }
    }
```

Output
count
count
count
count

* Example of for loop.

```
public class forloop {
    public static void main(String[] args) {
        for(int count = 1; count != 5; count++){
            System.out.println(count);
        }
    }
}
```

Output

1
2
3
4

* Celsius to Fahrenheit program.

```
import java.util.scanner;

public class Celsius To Fahrenheit {
    public static void main(String[] args) {
        scanner in = new Scanner(System.in);
        float tempC = in.nextFloat()
        float tempF = (temp * 9/5) +35;
        System.out.println(tempF);
    }
}
```

Output

45
113.0