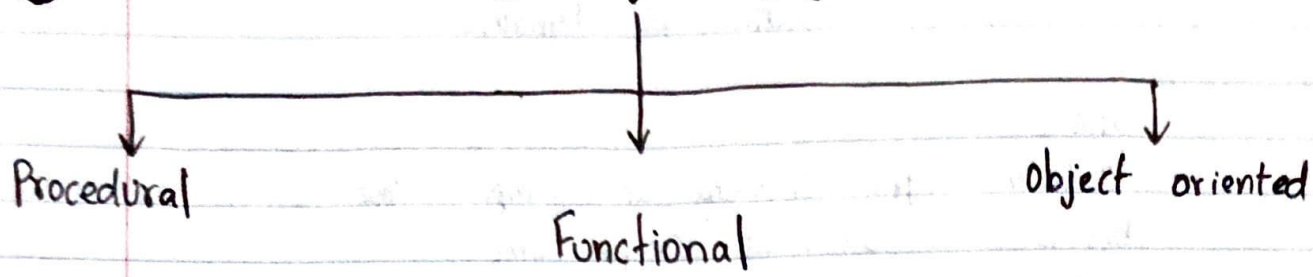
1/21/24

Types of Languages



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 in pure functions i.e. never modifies variables, but creates new ones as an output.
- used in situations where we have to perform lots of different operations on the same set of data, like ML
- First class functions

Object Oriented

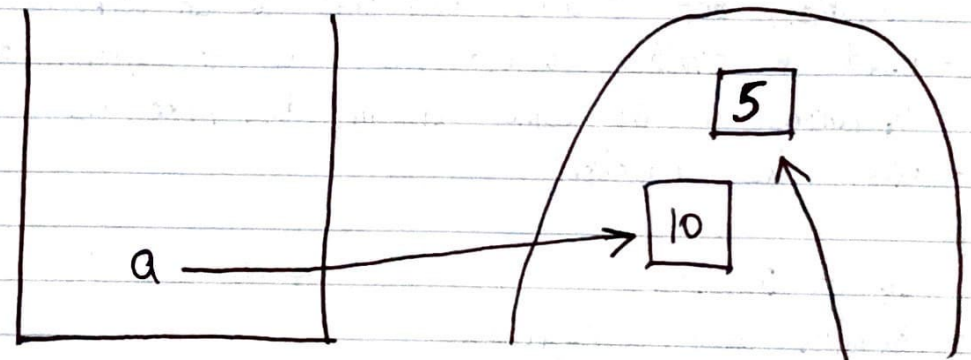
- Revolves around objects
- Code + Data = object
- Developed to make it easier to develop, debug, reuse, and maintain software

1/2/24

Static vs Dynamic

Static
- Perform type checking at compile time
- Errors will show at compile time
- Declare datatype before you use it
- More control

Dynamic
- Perform type checking at run time
- Error might not show till Program is run
- no need to declare datatype of variables
- Saves time in writing code but might give errors at runtime



Stack Memory

Heap Memory

☆ If it has no variable, it will be removed

a = 10

↓ Variable

↘ Object

☆ More than one variable can be used for an object

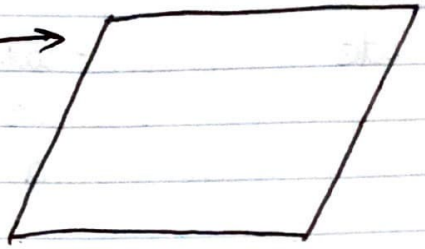
☆ If object is changed, it changes for all the variables

1/22/24

Flowcharts

Start / stop →

Input / output →



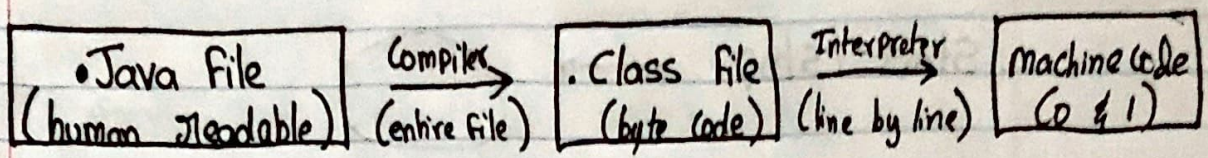
Processing →

Condition →

★ Psudeocode : A number of steps / Instructions written in human readable language / format.

1/23/24

How Java Executes Code



- Source code

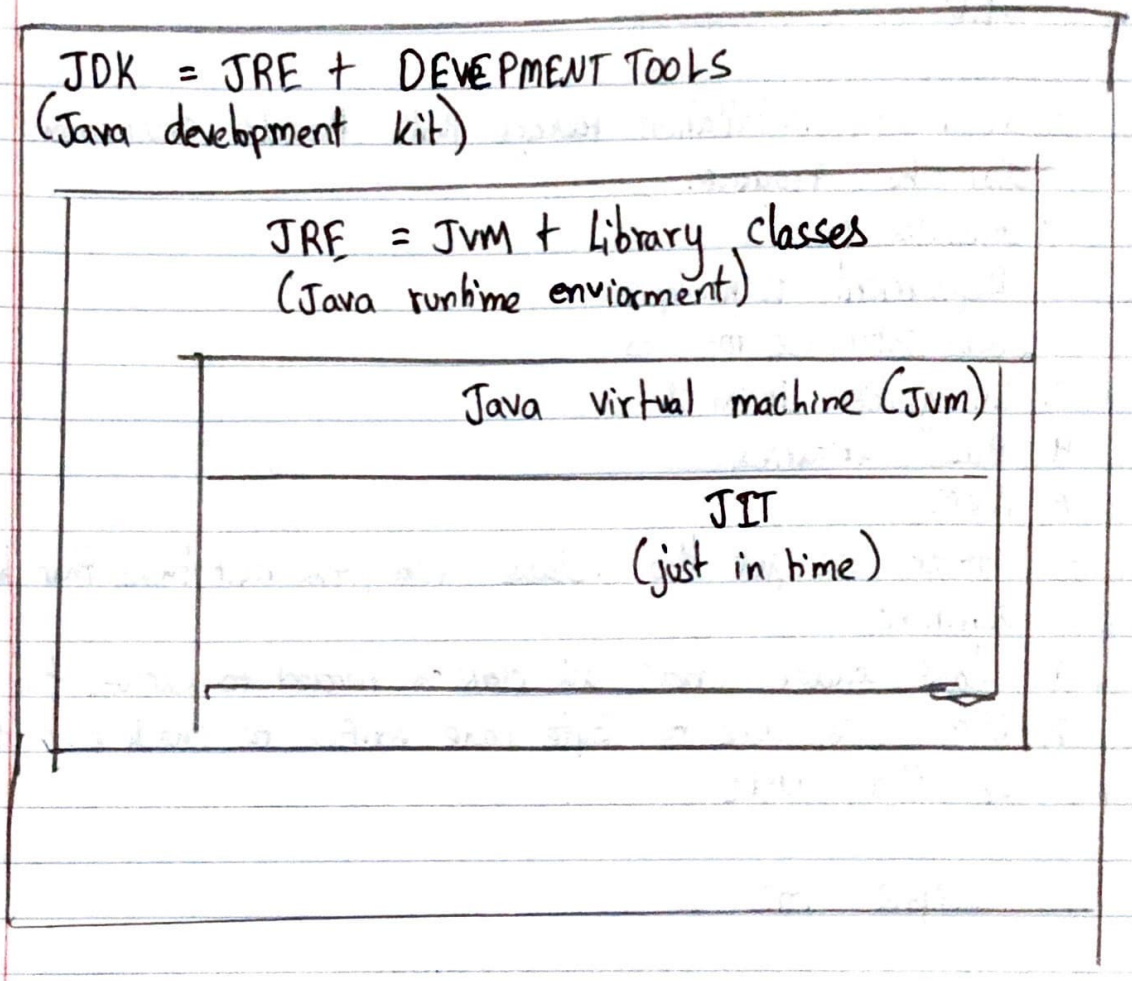
- Code does not directly run on a system
- needs Jvm to run
- reason why Java is Platform indipendent

Platform Indiperdence

- It means that the Byte code can run on any OS
- we need to convert Source code to machine code so that the computer can understand
- Compiler helps in doing this by turning it into executable code
- The executable code is a set of instructions for the computer
- After Compiling C/C++ code we get .exe file which is Platform dependent
- In Java we get bytecode, Jvm converts this to machine code
- Java is Platform dependent but Jvm is platform dependent

1/23/24

JDK VS JRE VS JVM VS JIT



JDK

- Provides an enviorment to develop and run Java Programs
- Is a Package that includes:

1. development tools - to provide an enviorment to develop your Programs
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 enviorment to only Run the Program
- Consists of:

1. Deployment technologies
2. User interface toolkits
3. Integration libraries
4. Base libraries
5. JVM

- After we get the .class files, the next thing that happens at Runtime:

1. Class loader loads all classes needed to execute the Program
2. Jvm sends code to Byte code Verifier to check for the format of the code

Compile time

.Java File

↓ Javac (compilation)

.class File

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 re-interpretation is not required

- makes execution faster
- Garbage collector

Runtime

(How Jvm works) Class Loader

- Loading :
  - reads .class file and generates binary data
  - an object of this class is created in heap

Class Loader

↓

Byte code Verifier

Linking :
- JVM verifies the .class file
- Allocates memory for class variables & default values
- Replace Symbolic References from the type with direct References

↓

Interpreter

↓

Initialization : all static variables are assigned with their values defined in the code and Static block

Runtime

↓

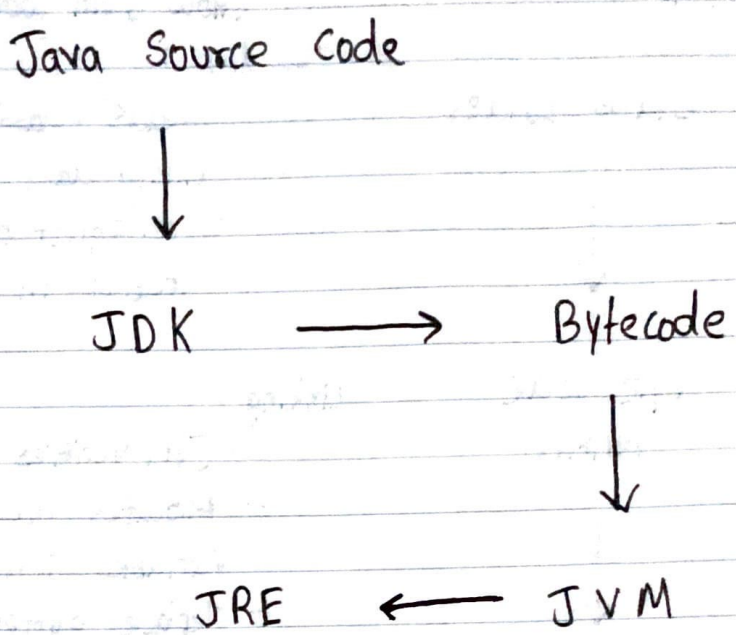
JVM Contains stack and Heap memory allocations.

↓

Hardware

1/23/24

How Java runs



1/24/24

JAVA

→ Classes should be written in uppercase in the first letter as it is a good practice

Keywords:

→ Public : Allows the class to be accessed from anywhere

→ Main function : Is a block of code from where your program starts

→ Function : Code which could be used again and again

→ static : To run the main function without creating objects of the class

→ void : The return type of the function. It doesn't allow a return value from the method.

→ string [] args : means an array of sequence of charecters that are passed to the main function.

→ Package : It is the folder where your java files lie. used for privacy when you don't want anyone to view your files

→ System : Allows you to access all the java files. contains class fields and methods

→ out : variable name of Printstream

→ Println : takes a string and outputs it

Scanner: text scanner which can pass primitive types and strings using regular expressions

Primitive data types: used to store / take input / output of a specific data type like integer, string. It cannot be broken further

Examples: int, char, float, double, long, boolean, string

Comments: used to put comments in code. Put two // to make a comment or /* */ to make longer comments.

☆ Reference variable is also called a Identifier
☆ object is also called a literal

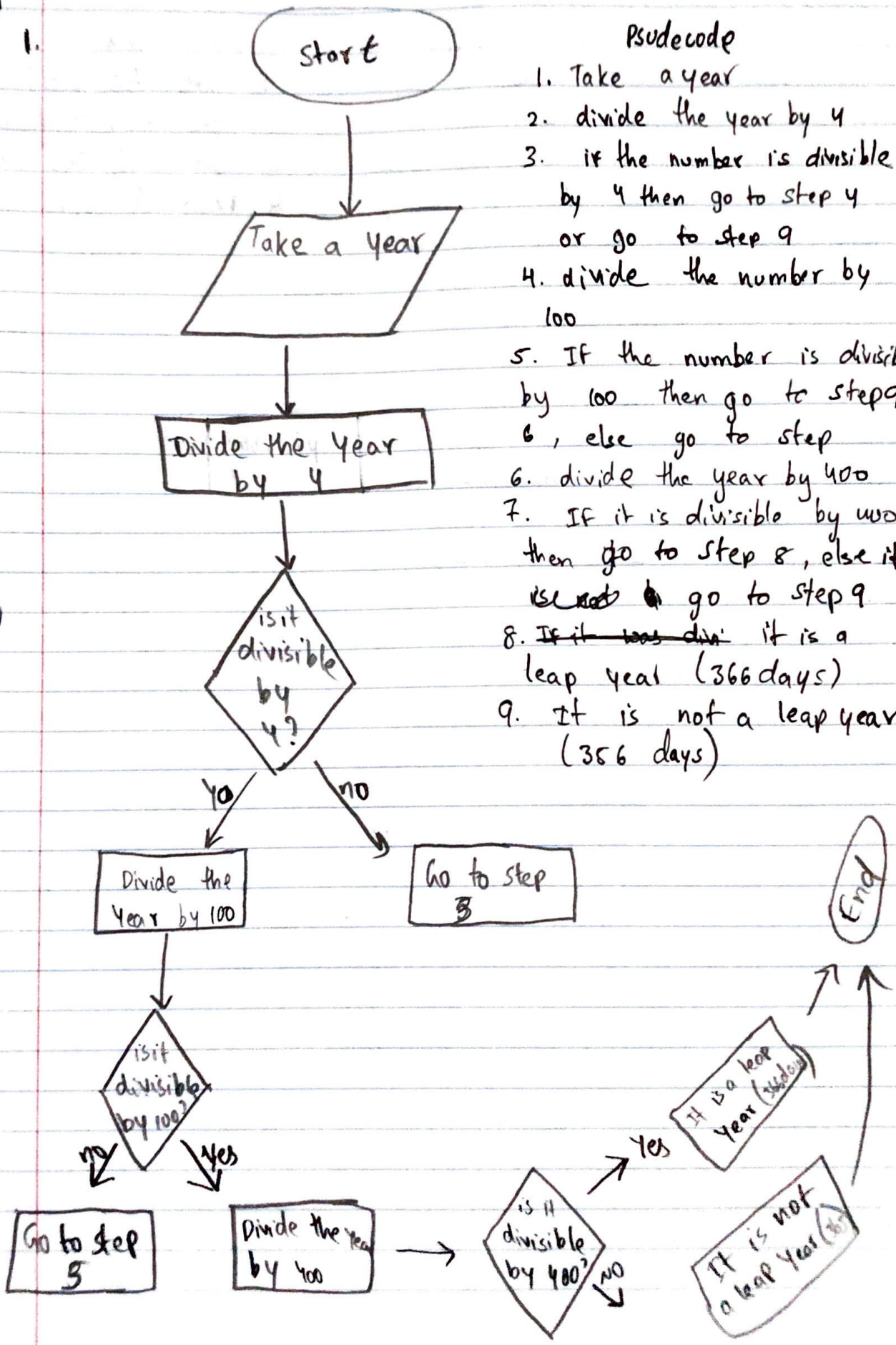
Type casting: Compresses the bigger number in a smaller ~~data~~ type

Unicode Principle: all languages can be put easily in Java.

~~Type Promotion~~ :

Assigment 1

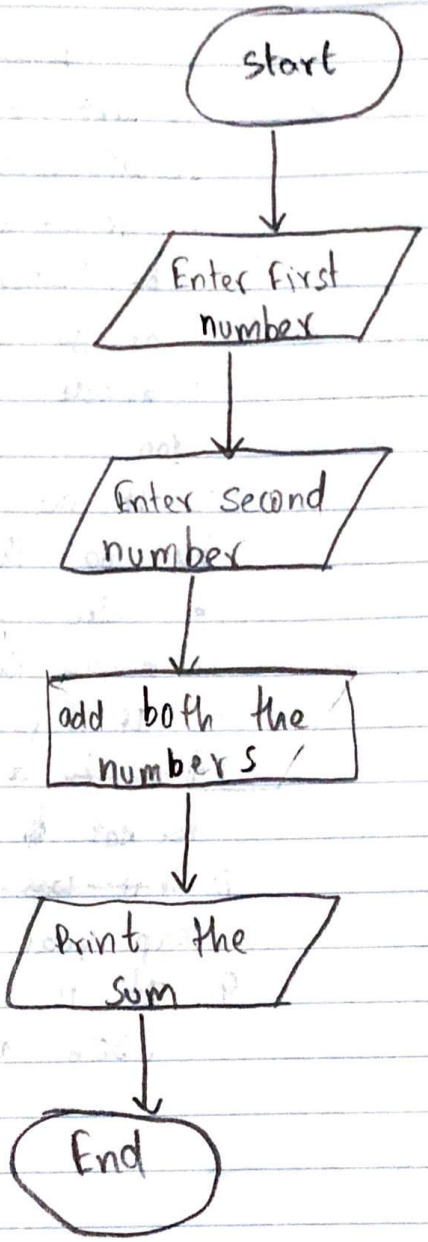
1.



Psudecode

1. Take a year
2. divide the year by 4
3. if the number is divisible by 4 then go to step 4 or go to step 9
4. divide the number by 100
5. If the number is divisible by 100 then go to step 6, else go to step
6. divide the year by 400
7. If it is divisible by 400 then go to step 8, else it ~~is not~~ go to step 9
8. ~~If it 400 div~~ it is a leap year (366 days)
9. It is not a leap year (356 days)

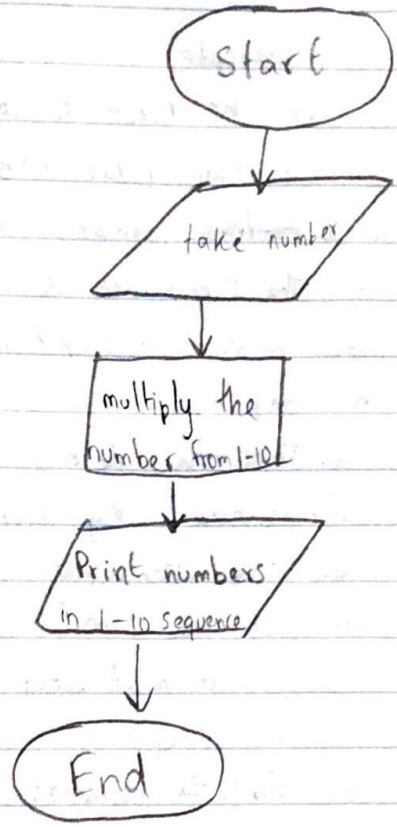
2.



Psudeocode

1. Take the first number
2. Take the second number
3. add both the numbers
4. Print the sum

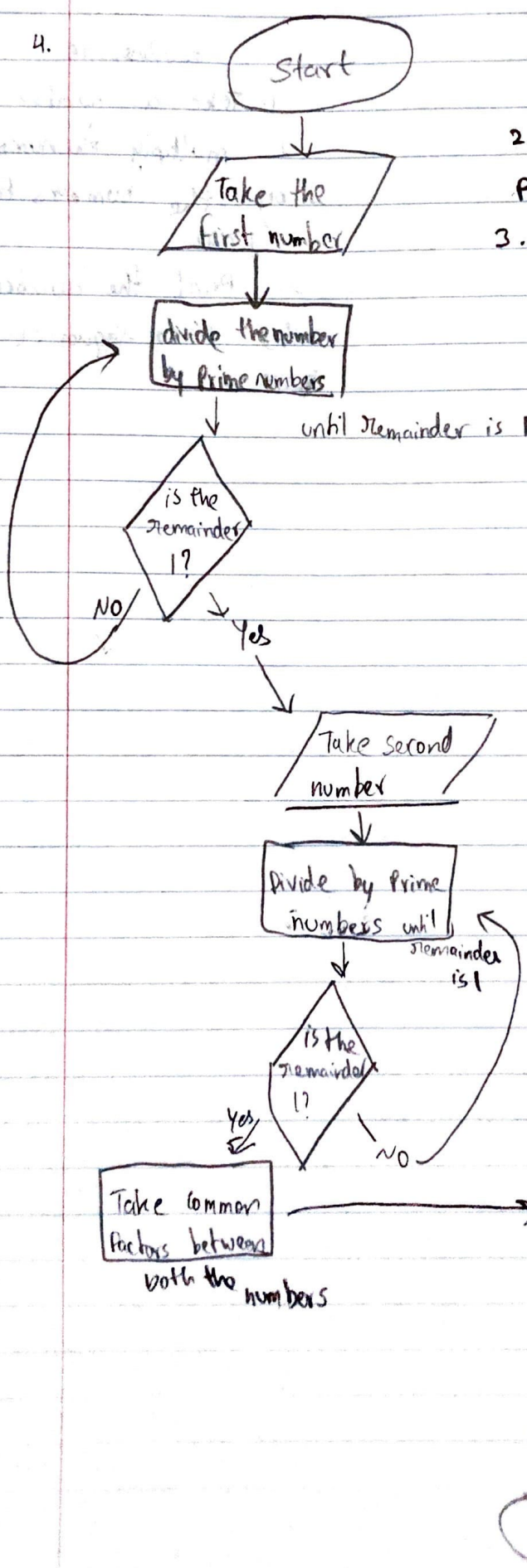
3.



Psudeocode

1. Take a number
2. multiply the number by the numbers from 1-10
3. Print the numbers in the 1-10 sequence

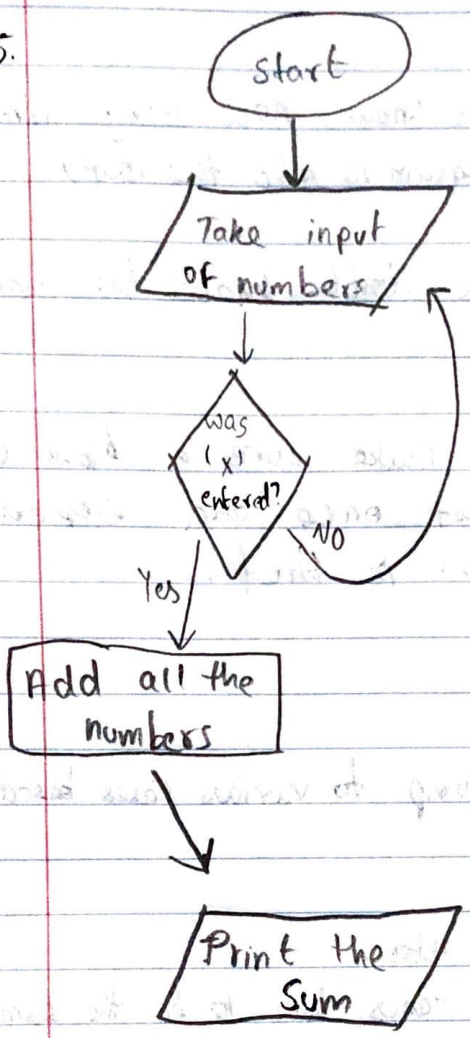
4.



Psudecode

1. Take the first number
2. divide the first number by Prime ~~factors~~ numbers until rem. is 1
3. If the remainder is 1, then move on to the next step else go to step 2
4. Take the second number
5. divide by Prime numbers till the remainder is 1
6. If the rem. is 1 then go to next step else do step 5
7. Take the common factors of both the numbers
8. multiply the common factors
9. Print the Product.

5.



Psudecode

1. Take input of numbers
2. was 'x' entered ? if yes then go to next step or else go to step 1
3. Add all the numbers
4. Print the sum

2/2/2024

Loops

1. For loop: used when you know how many times the Program will be run. (Program is also the loop)

2. While loop: used when you don't know how many times the loop will run

3. do while loop: used to make sure a block of code is executed at least once and repeated as long as the condition is met.

Switch case

In syntax statements, you can jump to various cases based on your expression.

Syntax:

```
switch (expression) {

    // cases
    case one:
      // do something
      break;

    case two:
      // do something
      break;

    default:
      // do something
```

Notes:
- cases have to be the same type of expressions, must be a constant or literal
- duplicate case values are not allowed
- break is used to terminate the sequence
- if break is not used, it will continue the next one
- default will execute when none of the above does
- if default is not in the end, put break after it.

New format

```
Switch (fruit) {
    case "mango" → sout ("King of fruits");
    "                                    "
    "                                    "
    default → sout ("Please enter a valid fruit")
```

Nested switch case : A switch case inside of a switch case.