1 .INTRODUCTORY TO PROGRAMMING

CLO 1

1. Explain various programming problem using design tools (C2)

At the end of the topic, student should be able to :

* **Define programme, programmer, programming language**
* **List the various types of programming language**
* **Explain the various types of programming language**

At the end of the topic, student should be able to :

- **Explain the types of programming**
- **Compare the types of programming**
- **List the stages involved in problem solving**
- **List the elements of problem analysis**
- **Explain the elements of problem analysis**

At the end of the topic, student should be able to :

- **Determine input, process and output for a given problem**
- **List the design tools for problem solving**
- **Explain the flowchart, psudo code and IPO chart**
- **Illustrate the various design tools**

At the end of the topic, student should be able to :

* Use the design tools to solve a given problem
* Write programme source code
* Execute the debugged programme source code
* Identify the various types of error in programming
* Explain the various types of error in programming

1.0 INTRODUCTORY TO PROGRAMMING

1.1 Introduction To Programming Concept And Terminology

1.1.1 Define Terms

a. Programme

a sequence of instructions that a computer can interpret and execute

b. Programmer

a person who designs and writes and tests computer programs

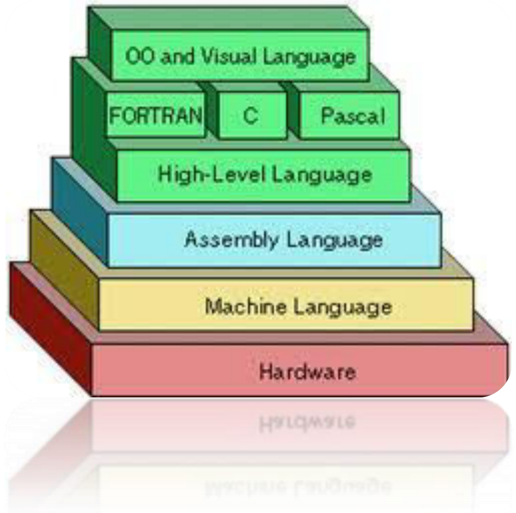
c. Programming language

programmers use to develop applications, scripts, or other set of instructions for a computer to execute

1.1.2 Types Of Programming Languages

i. Low Level Language

ii. High Level Language



1.1.3 i. Low Level Language

- Programs are written based on the internal architecture of machines

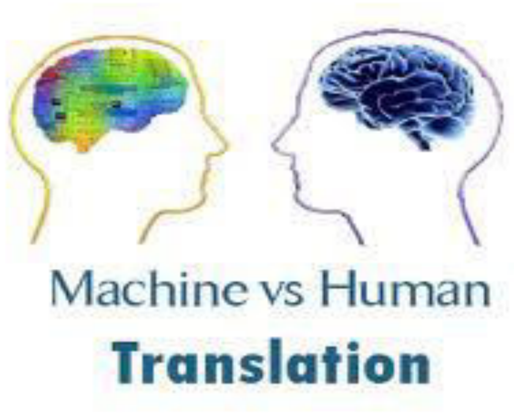
- Programmers must have extensive knowledge of the machine hardware

2 types :

* Machine Language
* Assembly Language

Machine Language

- Language that computers can understand.
- Instructions in this language are in the form of 1s and 0s.



Machine Language

* Examples:

  10100000100011001101 - Adding two numbers.

  10100010000011001101 - Subtracting two numbers.

Advantages :

* The computer processes the instructions in machine language very quickly

Disadvantages :

- Programmers must have knowledge of the machine hardware and its configuration.
- Programmer needs to remember a number of binary codes to write machine language programs.
- Machine language programs are very difficult to debug.

Assembly Language

Uses symbolic instructions (mnemonics) to represent the instructions in the programs.

```
 mov
 cld             es,[cs:Bu
PrintLoop:
 mov             dl,[es:si]
 inc             si
 mov             ah,2
 int             21h
                 si,BuffSize
 cmp             @@NotAtEnd
```

Assembly Language

Examples:

ADD R1 R2 – Adds two numbers stored in registers R1 and R2.

SUB R1 R2 – Is the instruction in assembly language to subtract two numbers stored in the registers R1 and R2.

Advantages :

- It is easy for programmers to remember the alphanumeric codes than the binary codes.
- Debugging is very easy when

  compared to machine language.

Disadvantages :

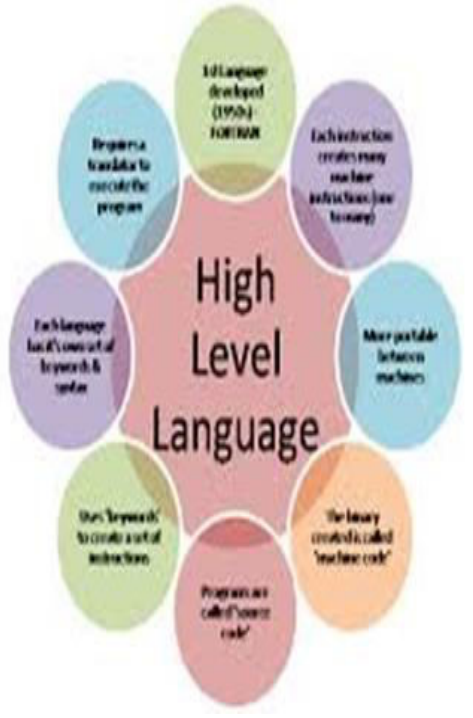
* The major disadvantage is that, it is machine dependent.

ii. High Level language

Languages that use English words and mathematical symbols for writing programs.

Java, C, C++, Basic,

Fortran


High Level Language
1st Language developed (1950s) FORTRAN
Each instruction creates many machine instructions (one to many)
More portable between machines
The binary created is called 'machine code'
Programs are called 'source code'
Uses 'keywords' to create a set of instructions
Each language has it's own set of keywords & syntax
Requires a translator to execute the program

ii. High Level language

Examples:

a + b – Adds the variable a and b.

a - b – Substrates the variable b from a.

Advantages :

- Programs are almost machine-independent.
- It is not necessary for the programmer to have knowledge about computer hardware.

Advantages :

- It is very easy to learn and write programs in
  high-level languages

Disadvantages :

- Programs written in high-level languages are slower in execution than the programs written using low-level languages.

1.1.4 Types of Programming

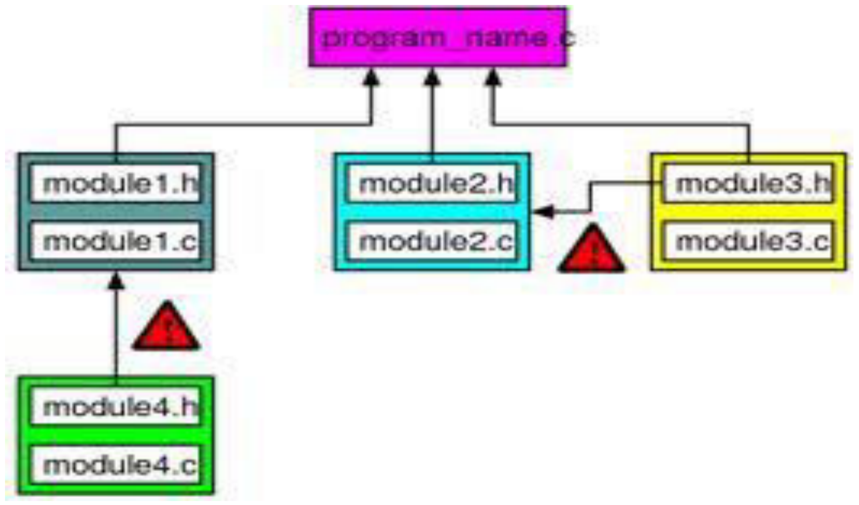
a. Structured Programming

Programming methodology in which the instructions are written in a sequence.

b. Modular Programming

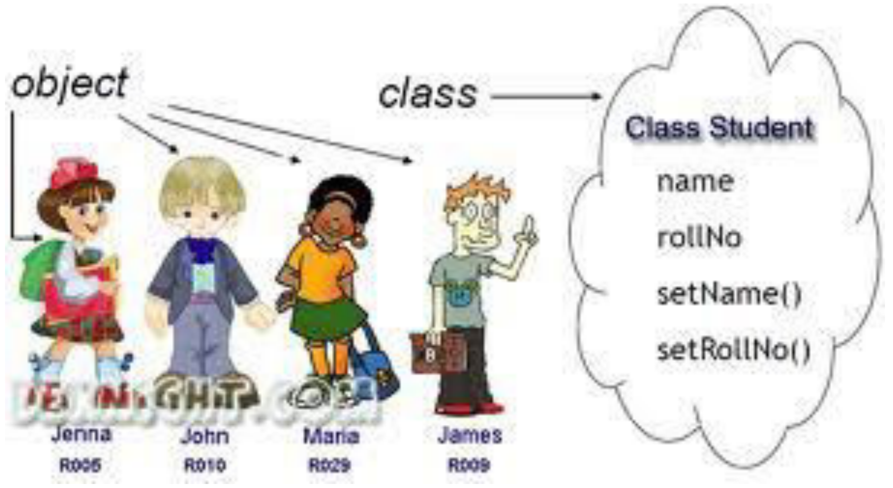
- Programming methodology in which the complex program is broken into number of simple modules.
- A module is an independent segment of the program that performs a specific task.

* When compared to structured programming, writing and debugging modular programs are easy.



c. Object-Oriented Programming

Programming methodology in which the data and the code are treated as a single unit.



1.1.5 Comparison Between Types Of Programming

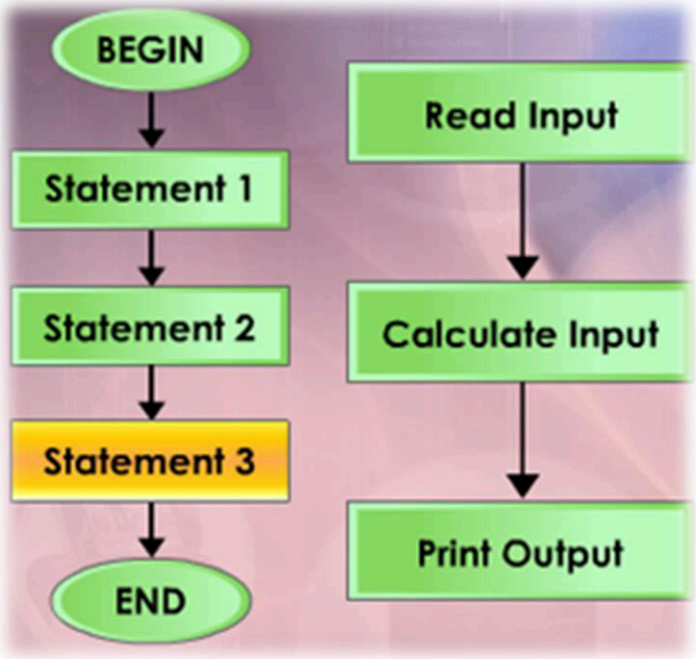
a. Structured Programming

b. Modular Programming

c. Object-Oriented Programming

Sequence Structure

- In sequence structure, any task or instruction leads to the next in a predetermined order.
- No task is skipped in the sequence.



Sequence Structure

Example:

Step1: Plug in the power cable.

Step2: Switch on the power.

Step3: Switch on the television Power button

Step4: Set the required channel using the remote control.

Activity 1a

Identify and arrange the following in the correct sequence:

1. **To work on a computer**

   a. **Work on the system.**

   b. **Switch on the system.**

   c. **Switch off the system.**

Activity 1b

Identify and arrange the following in the correct sequence:

2. To send a mail
   a. Sign out.
   b. Compose the mail.
   c. Send the mail.
   d. Sign in.

Activity 1c

Identify and arrange the following in the correct sequence:

3. To create a text file
   a. Type the content.
   b. Open the Notepad application.
   c. Save the file.

Activity 1d

Identify and arrange the following in the correct sequence:

4. To find the product of two numbers
   a. Calculate the product.
   b. Display the result.
   c. Get the input.

Activity 1e

Identify and arrange the following in the correct sequence:

5. To find the average of three numbers
   - a. Calculate the sum.
   - b. Accept the numbers.
   - c. Divide the sum by 3.
   - d. Display the result.

$$\text{Average} = \frac{\text{sum of the numbers}}{\text{number of addends}}$$

Assignment

1. Define sequence structure?

Selection Structure

- In selection structure, the instructions are executed based on a stated condition.
- Only one of the two tasks will be executed.

Selection Structure

Example:

```
if it rains
    Stay at home
        else
            Go out and play
```

Activity 2a

1. **When there is power, which of the two actions will take place?**

if there is power

Go by lift

else

Go by stairs

Activity 2b

2. When you have not completed your homework, which of the two tasks will be performed?

if you finish your homework

Watch movie

else

Go to study

Activity 2c

3. Today is Chinese New Year and hence it is a public holiday. Then, which task will be preformed?

```
if it is a holiday
        Need not go to school
    else
            Go to school
```

Activity 2d

4. When you come first in the race, which action will take place?

if you win in a race

Get a gold medal

else

Practise well and try again

Activity 2e

5. **When you enter an incorrect pin number in ATM, which of the two actions will take place?**

if you enter a valid ATM pin number

You can withdraw money

else

You cannot withdraw money

Assignment

1. Define selection structure?

Looping Structure

- In a looping structure, a set of instructions is executed several times based on a specific condition.

Looping Structure

Example

```
student_number = 1
When student_number less
than or equal to 20
    Display Grade
    student_number = student_number + 1
```

Activity 3a

1. How many times will the display statement get executed?

```
x=2
While x<5
        Display "Welcome"
        x=x+1
```

Activity 3b

2. You have ten 20 sen coins in hand. You need to pay a ringgit to your friend. How many times the following loop will be executed?

```
amount=0
While amount less than 1 ringgit
        Pay a 20 sen coin
        amount = amount + 20
```

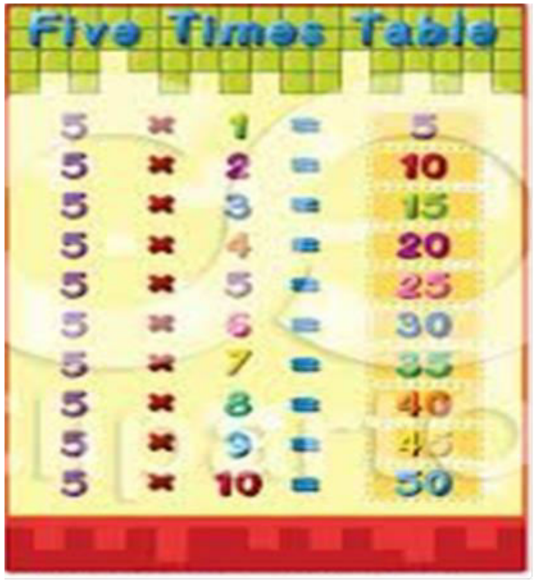
Activity 3c

3. To display the multiplication tables for 3 (till 3*10), the instructions will be as follows.

```
number =3
value =1
While value < 11
    Display number * value
        value= value + 1
```

Activity 3d

4. Similarly, write the instructions to display the multiplication tables for 5 (till 5*10).



Five Times Table

5	×	1	=	5
5	×	2	=	10
5	×	3	=	15
5	×	4	=	20
5	×	5	=	25
5	×	6	=	30
5	×	7	=	35
5	×	8	=	40
5	×	9	=	45
5	×	10	=	50

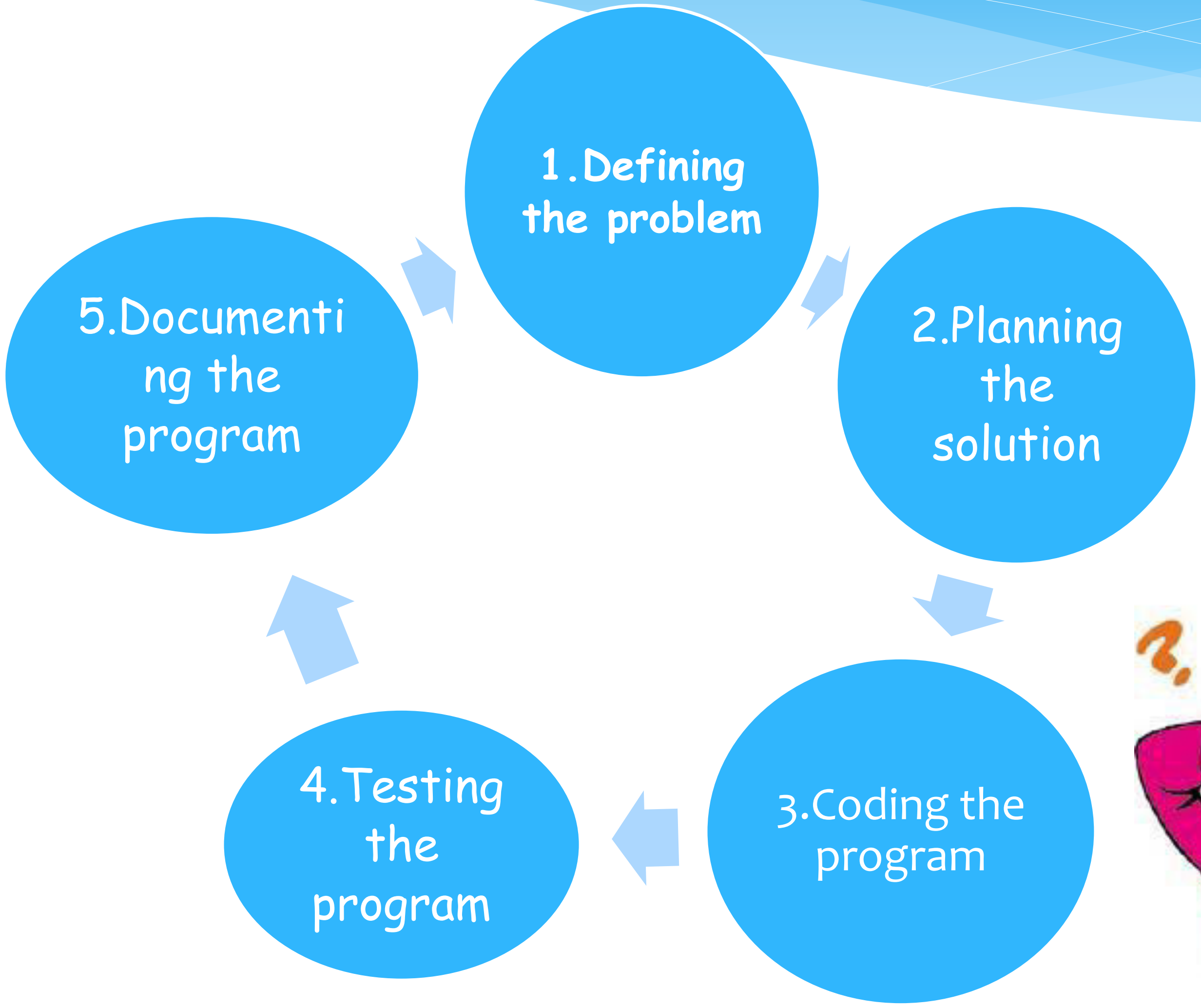
Advantages

Advantages of Structured Programming are:

- Easy to write
- Easy to understand
- Easy to modify

1.2 Understand Problem Solving

1.2.1 Stages in Problem Solving



1.2.2 Elements of Problem Analysis

a. Input
b. Process
c. Output

INPUT
PROCESS
OUTPUT

a. Input

information that must be acquired from the user before the problem can be solved

b. Process

the activity that takes place to determine the solution

c. Output

information to be displayed to the user indicating the result of the problem

1.2.4 Determine Input, Process and Output

e.g :

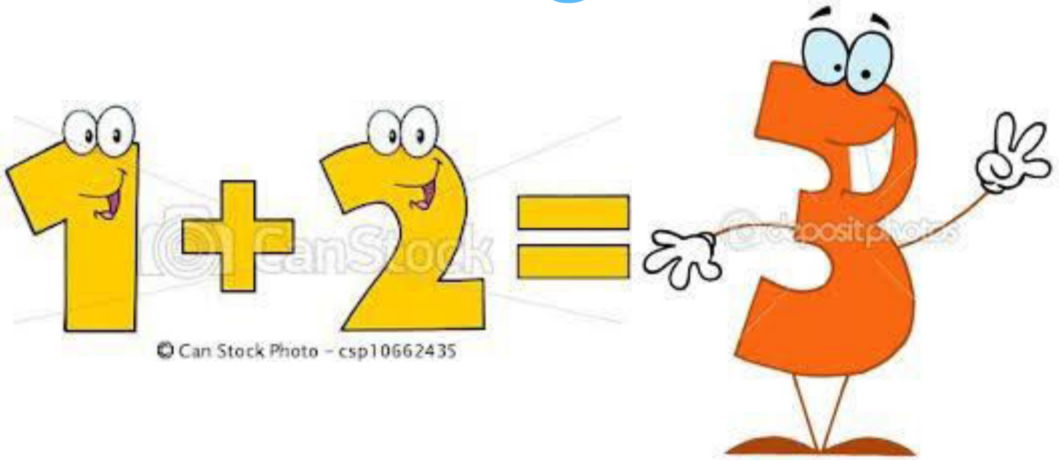
“Create a program to calculate sum of 2 integer numbers.”

1.2.4 Determine Input, Process and Output

input – 2 integer numbers

process –culculate sum (+)

output – sum of the 2 integer numbers

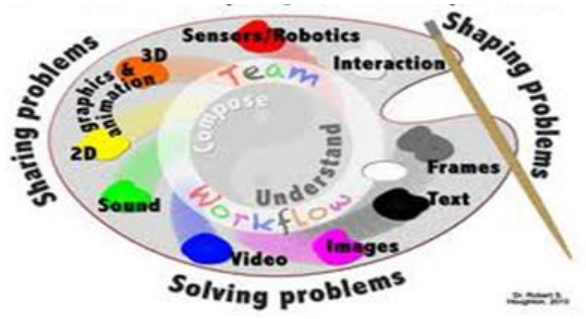


1.2.5 Design Tools For Problem Solving

a. Flowchart

b. Pseudo Code

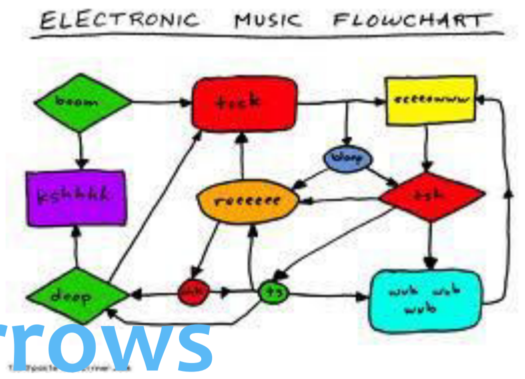
c. IPO Chart (Input-Process-Output Chart)



1.2.6 a. Flowchart

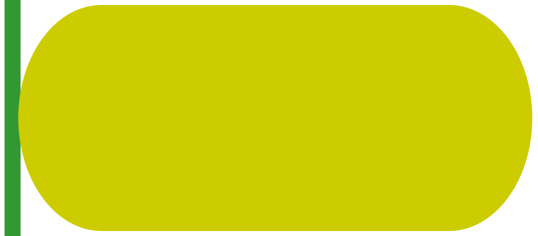
A flow chart is a graphical or symbolic representation of a process.

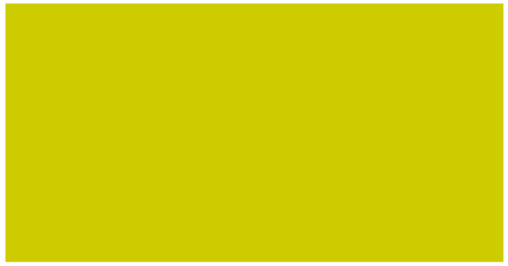
Each step in the process is represented by a different symbol and contains a short description of the process step.

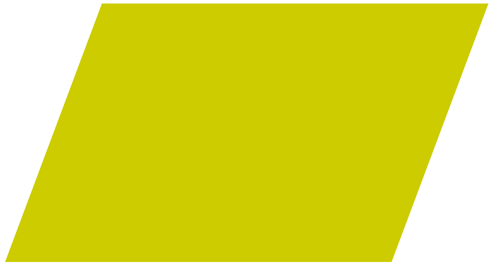


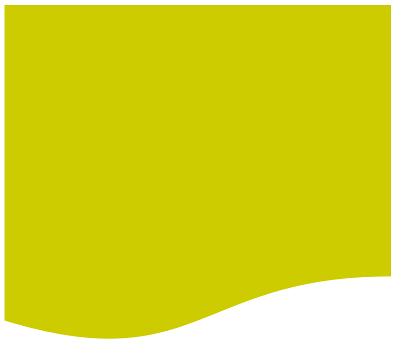
The flow chart symbols are linked together with arrows showing the process flow direction.

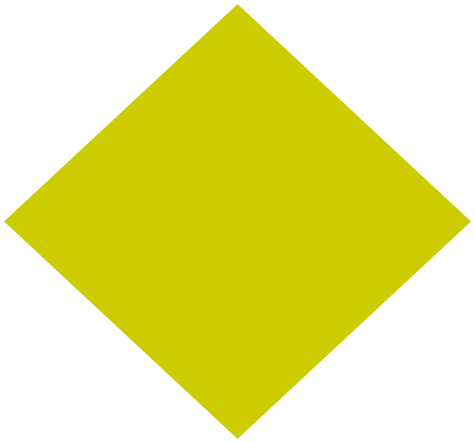
used in analyzing, designing, documenting or managing a process or program in various fields

Symbol	Name	Meaning
	Terminal (Start/End)	Used to represent the beginning (start) or the end (End) of a task

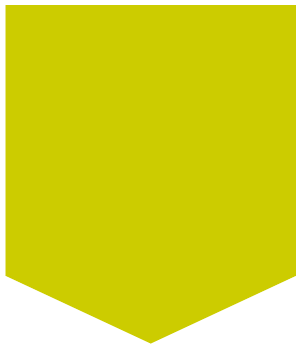
Symbol	Name	Meaning
	Processing	arithmetic and data-manipulation operations. The instruction are listed inside the symbol

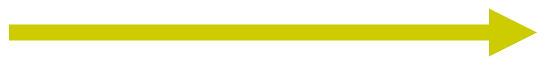
Symbol	Name	Meaning
	Input/output	Used for input and output operations, such as reading and printing. The data to be read or printed are describe inside

Symbol	Name	Meaning
	Output	Used output operations, such as printing the document. The data to be printed are describe inside

Symbol	Name	Meaning
	Decision	Used for any logic or comparison operations. has one entry and two exit paths. “YES” or “NO”

Symbol	Name	Meaning
●	Connector	Used to join different flow line

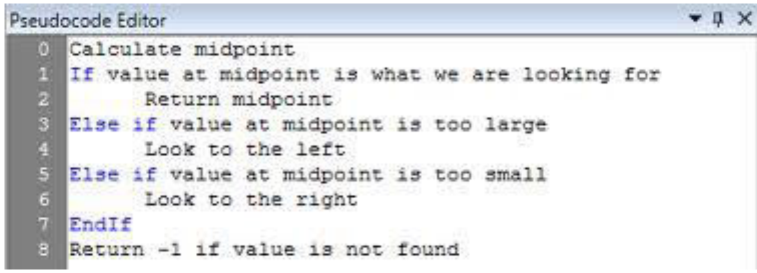
Symbol	Name	Meaning
	Offpage connector	Used to indicate that the flow chart continues to a second page

Symbol	Name	Meaning
	Flowline	Used to connect symbols and indicate the flow of logic

b. Pseudo Code

Pseudocode is a tool for planning, defining, or documenting the contents of a program routine or module. As the name implies, pseudocode is similar to (and often based on) real code.

They use special reserved words for expressing the logic of programs



```
Calculate midpoint
If value at midpoint is what we are looking for
    Return midpoint
Else if value at midpoint is too large
    Look to the left
Else if value at midpoint is too small
    Look to the right
EndIf
Return -1 if value is not found
```

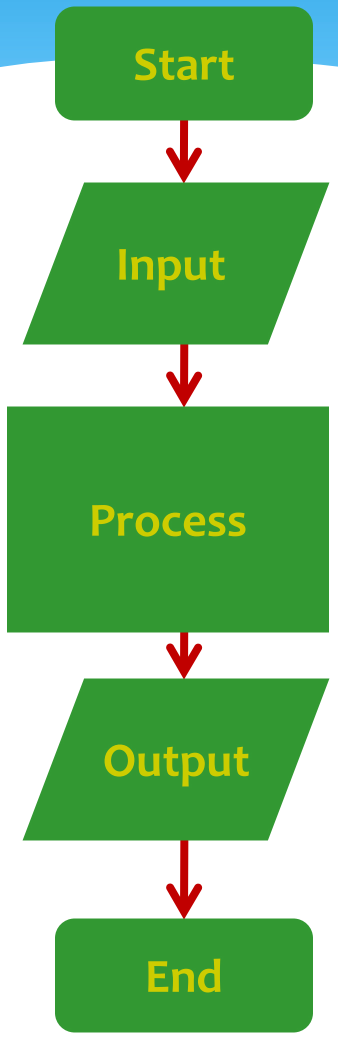
Keyword	Usage
//	Comment entry.
Begin….end	Specifies the block of statements that performs a specific task
Read	Input from record
Get	Input from keyboard
Put	Send to screen
Output	Send to screen
Display	Send to screen
If-Else	Specifies the condition and the block of statements that are executed based on the condition(selection)
Do-While	Specifies the condition and the block of statements that are executed based on the condition(repetition)

c. IPO chart(Input-Process-Output chart)

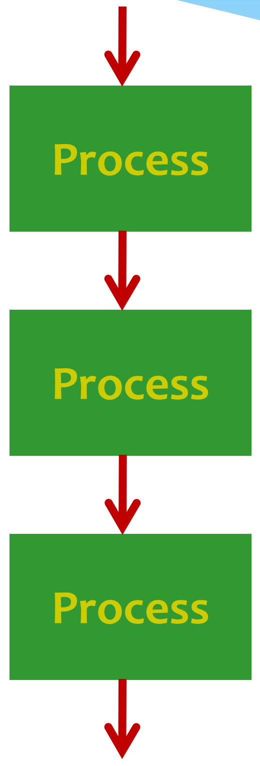
* identifies a program’s inputs, its outputs, and the processing steps required to transform the inputs into the outputs.

Input	Processing	Output

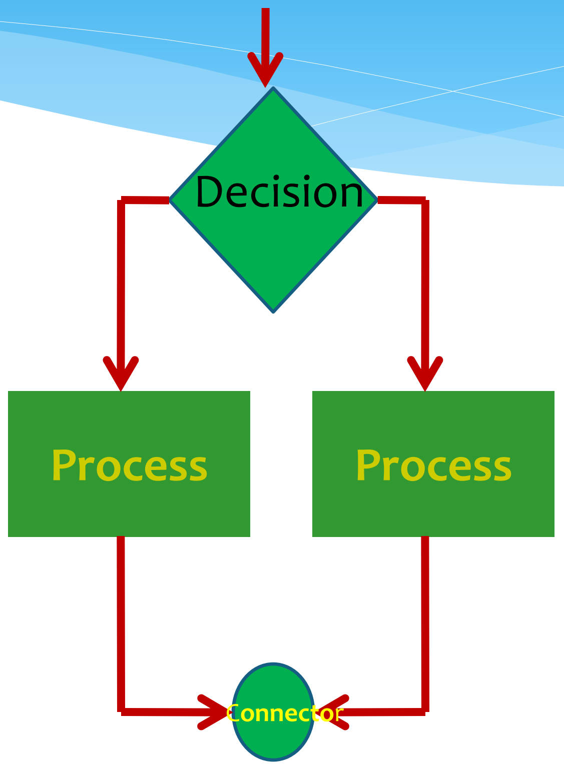
1.2.7 a. Flowchart



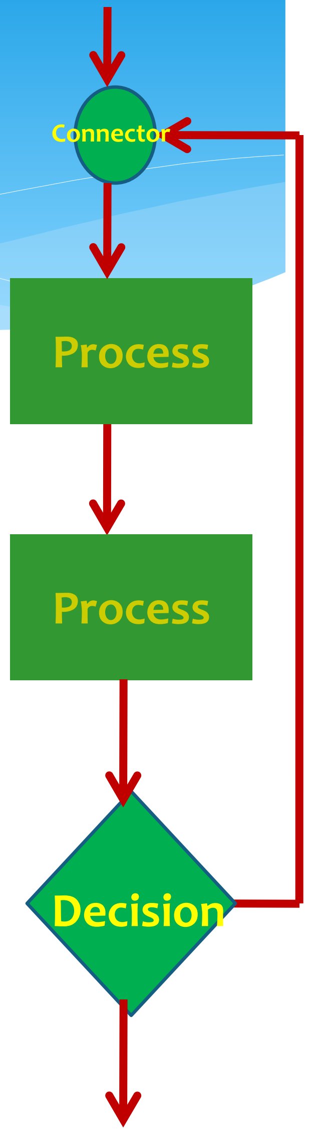
Basic Design



Sequence



Selection



Repetition

b. Pseudo Code

Basic	Selection	Repetition
Begin	Begin	Begin
Get data	Get data	Initialization
Process data	If ………	While ………
Display output	Display	Display Output
End	Else If	Increment……….
	Display else	End while
	End	End

c. IPO chart

Input	Processing	Output
		Ouput

Input	Processing	Output
Input		

Input	Processing	Output
Input	Process	Output

1.2.8 Use The Design Tools To Solve A Given Problem

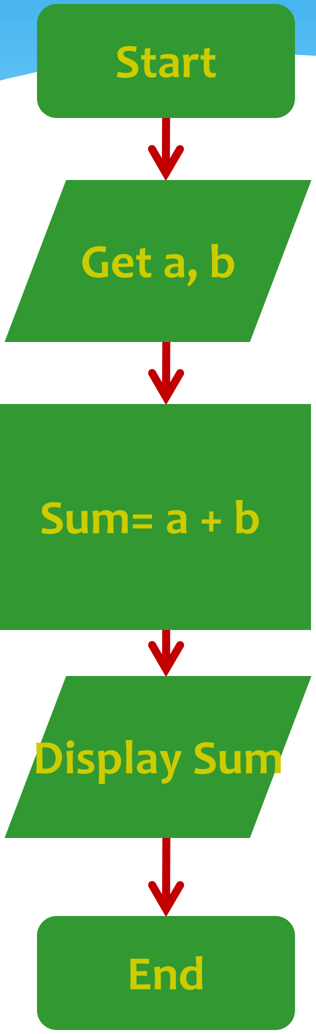
"Create a program to calculate sum of 2 integer numbers."

Input – 2 integer numbers

Process- adding 2 numbers

Output – Sum of 2 integer numbers

Flowchart   Pseudo Code   IPO Chart



```
Begin
Get data
Sum = a + b
Display output
End
```

Input	Processing	Output
		Sum

Input	Processing	Output
2 integer numbers		Sum

Input	Processing	Output
2 integer numbers	a + b	Sum

Exercise 1:

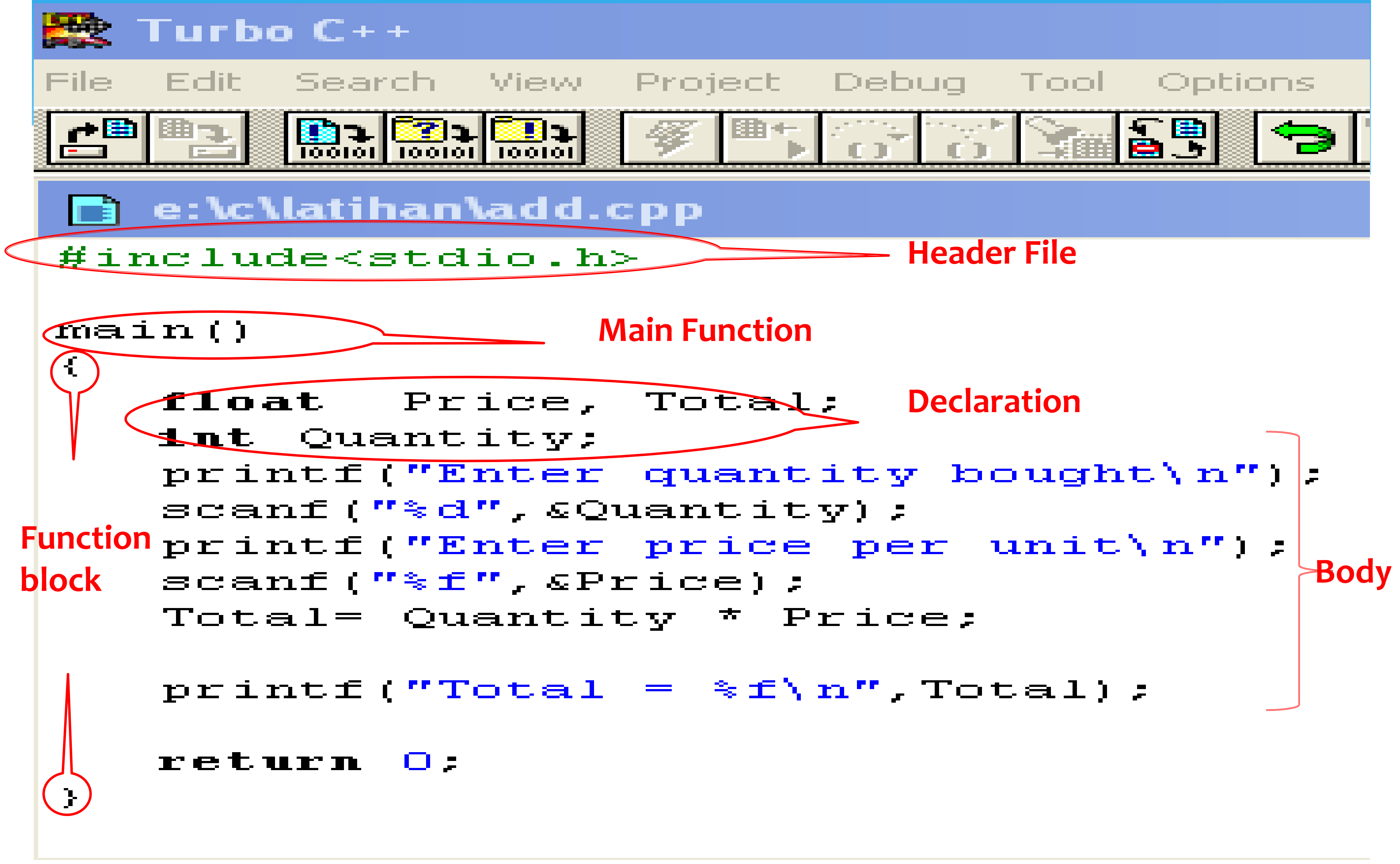
1. Write a pseudo code and draw a flow chart to calculate total for the item bought.

   Total=Price * Quantity

2. Write a pseudo code and draw a flow chart to calculate total mark if CA is 60% from total mark and FA is 40% from total mark

   Total Mark=(CA*0.6)+(FA*0.4)

1.2.9 Write Programme Source Code



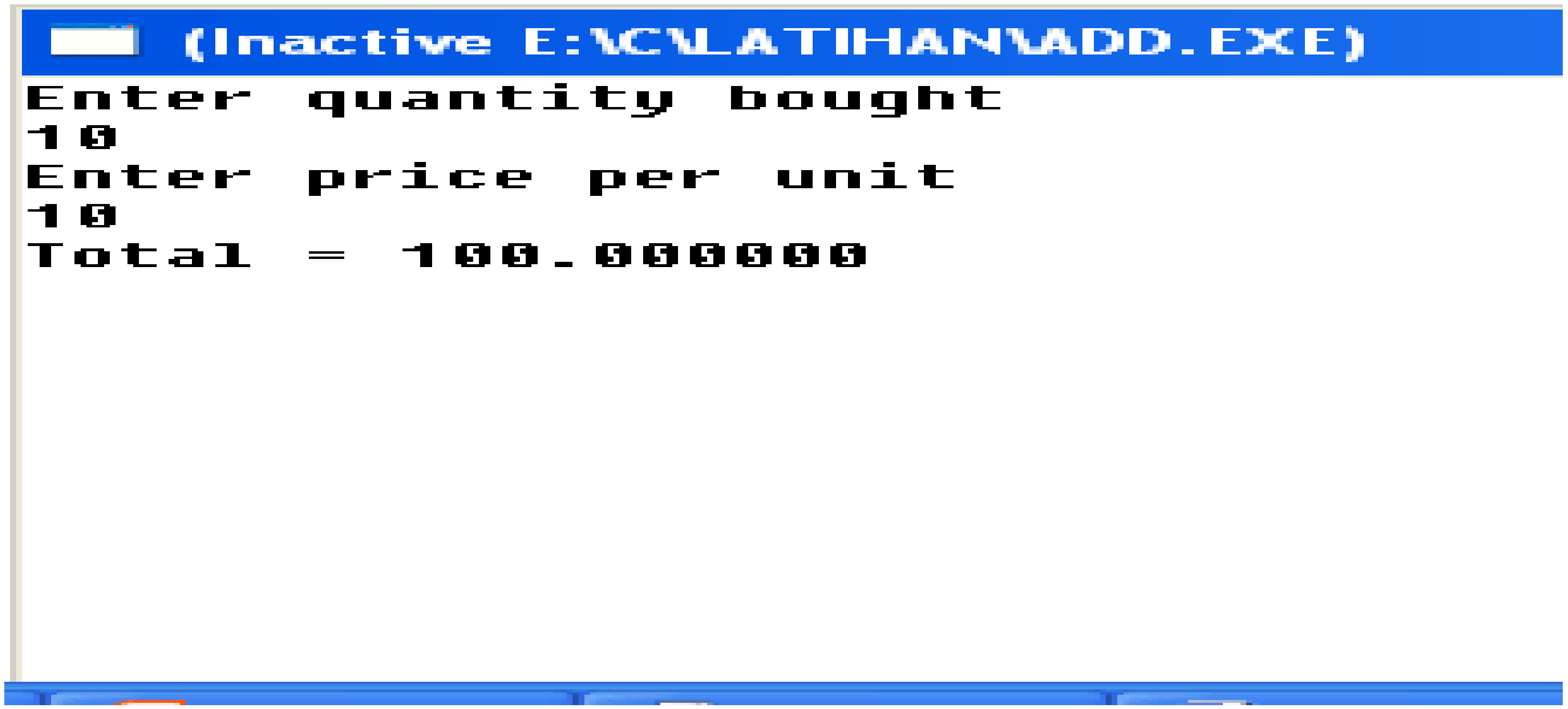
```
#include<stdio.h>

main()
{
     float   Price,   Total;
     int  Quantity;
     printf("Enter quantity bought\n");
     scanf("%d",&Quantity);
     printf("Enter price per unit\n");
     scanf("%f",&Price);
     Total= Quantity * Price;

     printf("Total = %f\n",Total);

     return 0;
}
```

1.2.10 Execute the debugged programme source code



```
(Inactive E:\C\LATIHAN\ADD.EXE)
Enter quantity bought
10
Enter price per unit
10
Total = 100.000000
```

1.2.11 Types of Error In Programming

a. Syntax Error

b. Run-Time Error

c. Logical Error

1.2.12 a. Syntax Error

- Syntax error is the one which occur if there is a violation in arrangement or in sequence of tokens subject to a language specific grammar

- Syntax error's will be trapped during compile time.

e.g :

```
void main()
{
printf(“hello World “)   /* ; Statement missing  */
}
```

b. Run-Time Error

runtime error is smart enough to get through the compilation process, and get unleashed during the execution

e.g :

```
void main()
{
            int i  = 10;
            int j = i / 0;

           printf(“%d”, j);
}
```

c. Logical Error

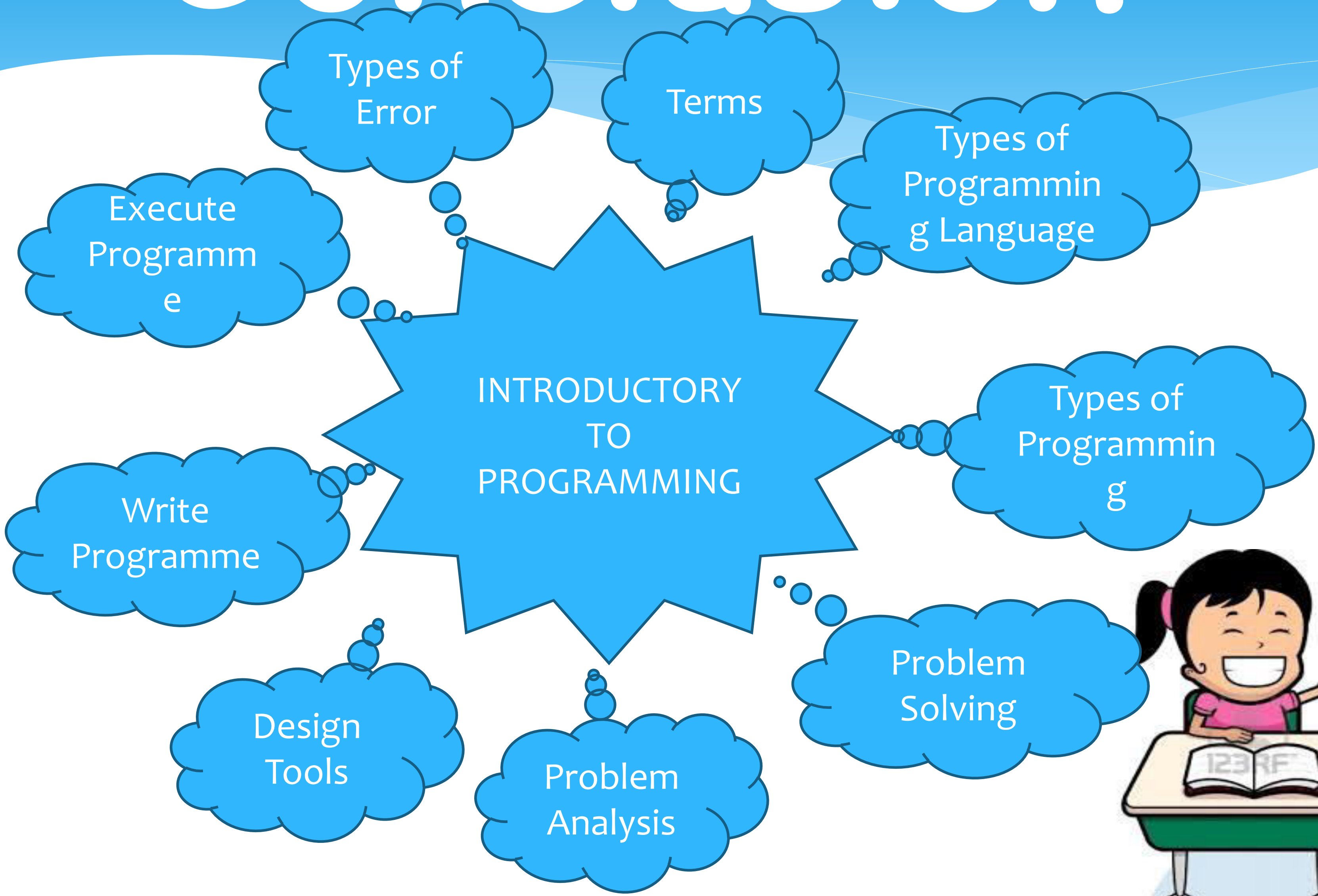
It may happen that a program contains no syntax or run-time errors but still it doesn't produce the correct O/P. It is because the developer has not understood the problem statement properly.

These errors are hard to detect as well.

e.g :

Error caused when any loop is not closed   at the right place.

Conclusion



INTRODUCTORY TO PROGRAMMING
Types of Error
Terms
Types of Programmin g Language
Execute Programm e
Types of Programmin g
Write Programme
Problem Solving
Design Tools
Problem Analysis

A+
123RF