

Symbol	Name
○	Terminal
—	Input / output
—	Process
◇	Decision
○	Connector
← ↓	Flow lines
—	Predefined Process

16-08-23

16-08-23

16-aug-23

CIT - 235

Wod

Practical No : 1

(Introduction to Flowchart)

- Terminal :-

It indicates the starting or ending of the program, Process or interrupt program.

- Input / output :-

It is used for any I/O operation.

- Process :-

The Process symbol indicates any internal operation inside the Processor.

- Decision :-

It is used to ask a question that can be answered as True/False

- Connector :-

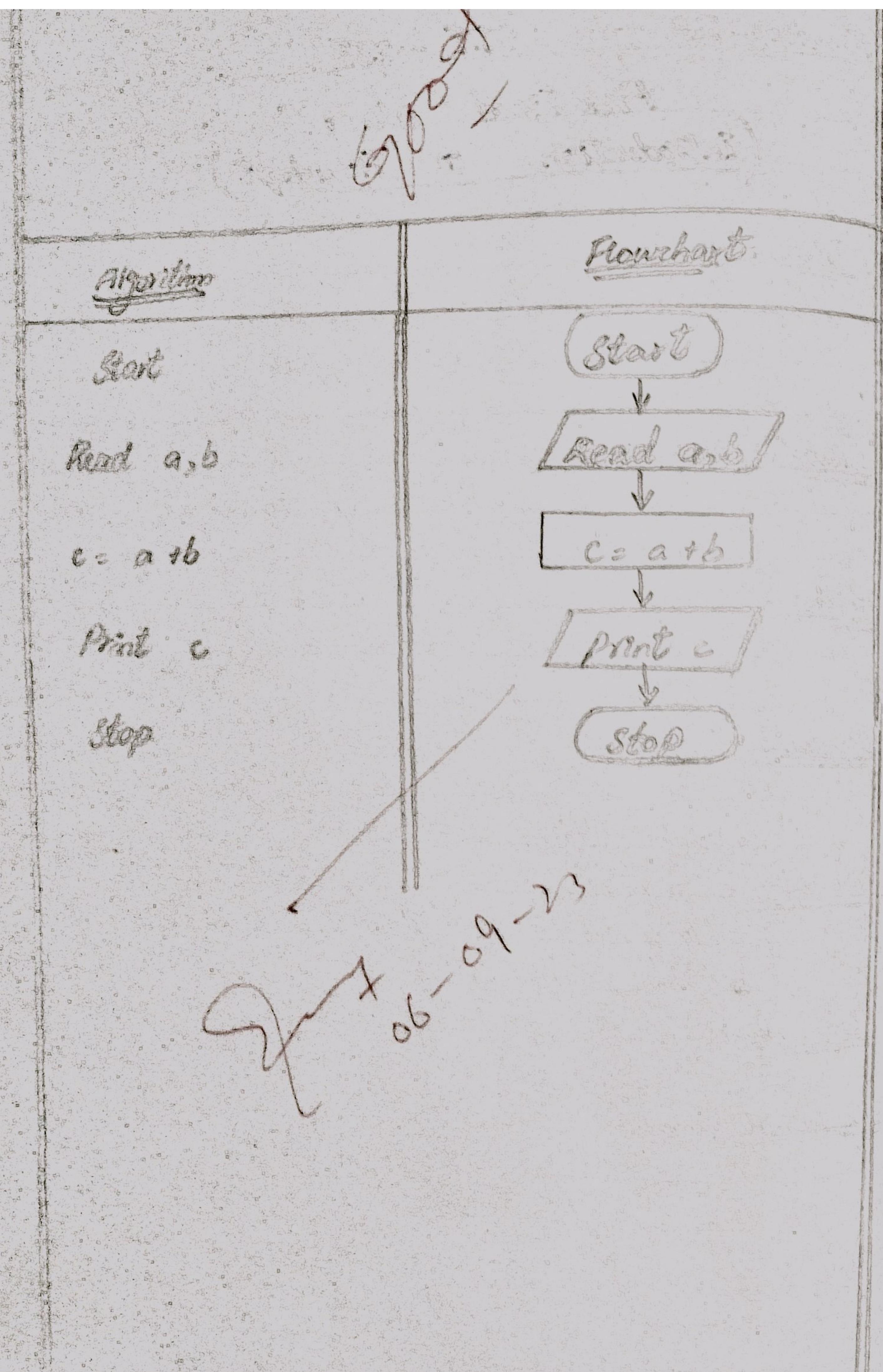
It allows the flowchart to be drawn without intersecting lines.

- Flow lines :-

Flow lines show direction of flow.

- Predefined process :-

It is used to invoke a subroutine or an interrupt program.



6-Sep-23 CIT-235 wed

Practical # 2
(Algorithm from flowchart)

Program :-

```

#include <stdio.h>
int main()
{
    int a, b, c;
    printf("Enter value of a:");
    scanf("%d", &a);
    printf("Enter value of b:");
    scanf("%d", &b);
    c = a + b;
    printf("Sum of given two numbers is : %d", c);
    return 0;
}
  
```

Digits	ASCII	Character	ASCII	Character	ASCII
0	48	A	65	a	97
1	49	B	66	b	98
2	50	C	67	c	99
3	51	D	68	d	100
4	52	E	69	e	101
5	53	F	70	f	102
6	54	G	71	g	103
7	55	H	72	h	104
8	56	I	73	i	105
9	57	J	74	j	106
		K	75	k	107
		L	76	l	108
		M	77	m	109
		N	78	n	110
		O	79	o	111
		P	80	p	112
		Q	81	q	113
		R	82	r	114
		S	83	s	115
		T	84	t	116
		U	85	u	117
		V	86	v	118
		W	87	w	119
		X	88	x	120
		Y	89	y	121
		Z	90	z	122

13-SEP-23

Microprocessor Architecture

wednesday

Practical # 3

(Applications of ASCII code)

Character	ASCII	64	32	16	8	4	2	1
R	82	1	0	1	0	0	1	0
A	65	1	0	0	0	0	0	1
J	74	1	0	0	1	0	1	0
A	65	1	0	0	0	0	0	1
A	65	1	0	0	0	0	0	1
B	66	1	0	0	0	0	1	0
D	68	1	0	0	1	0	0	0
U	85	1	0	1	0	1	0	1
R	82	1	0	1	0	0	1	0
R	82	1	0	1	0	0	1	0
A	65	1	0	0	0	0	0	1
H	72	1	0	0	1	0	0	0
E	69	1	0	0	0	1	0	1
E	69	1	0	0	0	1	0	1
m	77	1	0	0	1	1	0	1

Q 7

13-09-23

The Binary Number System :-

Bit value	1	1	1	1	1	1	1	1
Position value as a power of base 2	128	64	32	16	8	4	2	1
Bit number	7	6	5	4	3	2	1	0

20-sep-23

Microprocessor Architecture

wednesday

Practical # 4

(Introduction to Assembly language)

Q. What is Assembly language?

Each personal computer has a microprocessor that manages the computer's arithmetical, logical and control activities. Each family of processors has its own set of instructions for handling various operations like getting input from keyboard, displaying information on screen and performing various other jobs. These sets of instructions are called machine language instructions. Processor understands only machine language instructions which are strings of 0s and 1s. However machine language is too obscure and complex for using in software development. So the low level assembly language is designed for a specific family of processors that represents various instructions in symbolic code and a more understandable form.

Q. Advantages of assembly language:-

An understanding of assembly language provides knowledge of:

- Interface of programs with OS, Processor and BIOS;
 - Representation of data in memory and other external devices;
 - How Processor accesses and executes instructions;
 - How instructions access and process data;
 - How a program access external devices.
- Other advantages of using assembly language are:

The Hexadecimal Number System :-

Decimal number	Binary representation	Hexadecimal representation
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1001	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Good
out of 25

- It requires less memory and execution time;
- It allows hardware-specific complex jobs in an easier way;
- It is suitable for time-critical jobs;

Basic Features of PC Hardware :-

The main internal hardware of a PC consists of the processor, memory and the registers. The registers are processor components that hold data and address. To execute a program the system copies it from the external device into internal memory.

The processor executes the program instructions.

The fundamental unit of computer storage is bit; it could be on (1) or off (0). A group of more related bits makes a byte. So the parity bit is used to make the number of bits in a byte odd. If the parity is even, the system assumes that there had been a parity error which might have caused due to hardware fault or electrical disturbance.

The processor supports the following data sizes:

- Word : a 2-byte data item
- Doubleword : a 4-byte (32-bit) data item
- Quadword : an 8-byte (64-bit) data item
- Paragraph : a 16-byte (128-bit) area
- Kobyte : 1024 bytes
- Megabyte : 1,048,576 bytes.

DOS Box Commands :-

edit filename.asm
MASM filename.asm ;
Link filename.obj ;
file name.exe.

Program to print a single character :-

- DOS eg
• model small
• stack 100h
• data
• code

main program
↓
main end P
end main

Oct-23

(CIT-235)
microprocessor Architecture

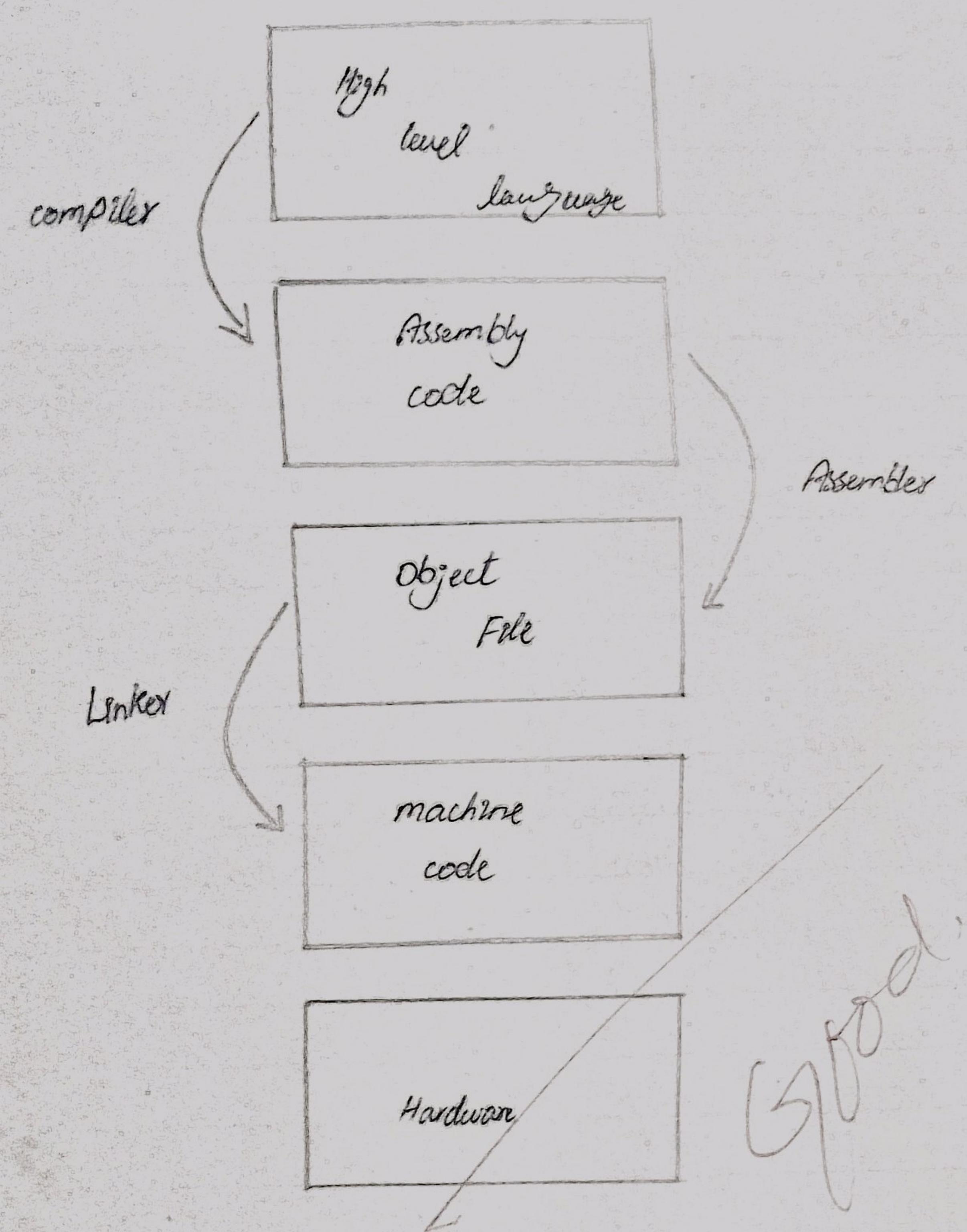
Wednesday

Practical # 05

(Familiarize Environment Setup)
Assembly language is dependant upon the instruction set i.e. architecture of the processors. In this tutorial, we focus on Intel 32 processor like pentium. To follow this tutorial, you will need:

- o An IBM PC or any equivalent compatible computer.
- o A copy of Linux operating system.
- o A copy of NASM assembler programs, like;
There are many good assembler programs, like:
 - o Microsoft Assembler (MASM).
 - o Borland Turbo assembler (TASM).
 - o The GNU assembler (GAS).
- o Tree. You can download it from various web sources.
- o well documented i.e. You will get lots of information on net.
- o could be used on both Linux and Windows.

DOS BOX Emulator :-



Installing NASM :-

If you select "Development Tools" while installed linux, you may NASM installed along with the linux operating system &, you don't need to download &, install it separately. For you checking whether you already have NASM installed, take the following steps :

- o open a Linux terminals
- o Type where is NASM &, Press **ENTER**
- o if it is already installed then a line like, `NASM: /user/bin/nasm` appears otherwise, you will see just `NASM` then you need to install NASM take the following step

9/10/23

```
Z:\> mount c c:\mp  
Z:\> c:  
C:\> edit file.asm  
C:\> masm file.asm;  
C:\> link file.obj;  
C:\> file.exe
```

18. oct - 23

Microprocessor Architecture (C17-235) wednesday

Practical # 06

"Program to print a single character"

```
dos seg  
.model small  
.stack 100h  
.data  
.code  
main proc  
mov dl, 'A'  
Mov ah, 2  
int 21h  
mov ah, 4ch  
int 21h  
main end p  
end main
```

9/10.23
18

```
Z:\> mount c c:\mp  
Z:\> c:  
C:\> editfile2.asm  
C:\> macrofile2.asm ;  
C:\> link file2.obj;  
C:\> file2.exe
```



1-Nov-23

CIT - 235

WKA

practical No : 07

3 Program to print a name
with character.

```
dosseg  
.model small  
.stack 100h  
.data  
.code  
main proc  
mov dl, 'A'  
mov ah, 2  
int 21h  
mov dl, 'L'  
int 21h  
mov dl, 'I'  
int 21h  
mov ah, 4ch  
main endp  
end main
```

Start
21h

CIT-235

(microprocessor Architecture)

5-Dec-23

Topic

Practical # 8

i program to print characters

from A to Z.

Dosseg

int 21h

```
Z:\> mount c c:\mp  
Z:\> C:  
C:\> edit file3.asm  
C:\> masm file3.asm  
C:\> link file3.obj;  
C:\> file3.exe
```

model small	mov dl, 'P'	int 21h
stack 100h	int 21h	mov ah, 2
data	mov dl, 'H'	int 21h
, code	mov ah, 2	mov dl, 'A'
main proc	int 21h	mov ah, 2
mov dl, 'R'	mov dl, 'I'	int 21h
mov ah, 2	mov ah, 2	mov dl, 'R'
int 21h	int 21h	mov ah, 2
mov dl, 'B'	mov dl, 'J'	int 21h
mov ah, 2	mov ah, 2	mov dl, 'S'
int 21h	int 21h	int 21h
mov dl, 'C'	mov dl, 'K'	mov dl, 'T'
mov ah, 2	mov ah, 2	mov ah, 2
int 21h	int 21h	int 21h
mov dl, 'D'	mov dl, 'L'	mov dl, 'U'
mov ah, 2	mov ah, 2	int 21h
int 21h	int 21h	mov dl, 'W'
mov dl, 'T'	mov dl, 'M'	int 21h
mov ah, 2	mov dl, 'N'	mov dl, 'X'
int 21h	int 21h	int 21h
mov dl, 'F'	mov ah, 2	mov dl, 'Y'
mov ah, 2	int 21h	int 21h
int 21h	mov dl, 'O'	mov dl, 'Z'
	mov ah, 2	wained P
		end main

```
Z:\> mount C C:\MP  
C:\> drive mounted as local  
directory  
  
Z:\> C:  
C:\> edit file5.asm  
C:\> masm file5.asm;  
C:\> link file5.obj;  
C:\> file.exe;
```

01-24

CIT-235

Tuesday

Practical # 09
"Program to multiply two numbers."
; program to multiply two numbers.

Dosseg

model small

stack 100h

data

code

main proc

mov al, 5

mov bl, 2

MUL bl

ASM

mov ch, ah

mov cl, al

mov dl, ch

add dl, 48

mov ah, 2

int 21h

mov dl, cl

add dl, 48

mov ah, 2

int 21h

mov ah, 4ch

int 21h

main ends

end main

23 | 01

24

```
Z:\> mount c c:\MP  
C:\> drive mounted as  
local directory  
  
Z:\> C:  
C:\> edit file4.asm  
C:\> masm file4.asm ;  
C:\> link file4.obj ;  
C:\> file4.exe ;
```

30-01-24

CIT - 235

Tues

Practical # 10

"Program to subtract two numbers"

; program to subtract two numbers
DOSBox

- model small
- stack 100h
- data
- code

main proc

mov bl, 3

mov cl, 1

sub bl, cl

add bl, 48

mov dl, bl

mov ah, 2

int 21h

mov ah, 4ch

int 21h

main endp

end main

81

30-01-24