ASSEMBLY LANGUAGE

SONIYA SHARMA

Institute of Law, Jiwaji University Gwalior(M.P.)

Email-sharma.soniya845@gmail.com

Class:Bcom LLB VI Sem, BA LLB VI Sem

SUBJECT: Computer application

UNIT:II

TOPIC:Assembaly Language

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 such as getting input from keyboard, displaying information on screen and performing various other jobs. These set of instructions are called '**machine language instructions**'.

- A processor understands only machine language instructions,
- which are strings of 1's and 0's.
- 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.

Advantages of Assembly Language

- **It's easy to understand the following function of a system with the minimum knowledge of assembly language-**
- How programs interface with OS, processor, and BIOS;
- How data is represented in memory and other external devices;
- How the processor accesses and executes instruction;
- How instructions access and process data;
- How a program accesses external devices.

Other advantages of using assembly language are

- It requires less memory and execution time;
- It allows hardware-specific complex jobs in an easier way;
- It is suitable for time-critical jobs;
- It is most suitable for writing interrupt service routines and other memory resident programs.

- Sometimes referred to as **assembly** or **ASM**, an **assembly language** is a low-level programming language.
- Programs written in assembly languages are compiled by an assembler. Every assembler has its own assembly language, which is designed for one specific computer architecture.
-

Example of assembly language

- Print :
- **Hello World!** In 2 bit assembly for window

- Here is "Hello, World" written for a 32-bit Intel processor. It will also run on a 64-bit processor. We will compile and run it on Windows 10.

```
 global  _main
    extern  _printf
    section .text
_main:
    push    message
    call    _printf
    add     esp, 4
    ret
message:
    db  'Hello, World!', 10, 0
```

- To begin, open Notepad. Copy and paste the code above into a new text file, and save the file as *hello.asm*.

- To compile the assembly, we use **NASM**, the Netwide Assembler. It can be downloaded at the NASM site.