University of Diyala

College of Engineering

Department of Computer Engineering

Computer Science

1st stage

Lecture #3

- Course Number: U 102
- Course Name: Computer Science
- Credit Hours: (2-1-0-2)
- Prerequisites: None
- Course Contents: Computer Architecture, Computer Assembly and parts Characteristics, History of Computer, Generations of computer, Types of computer, Personal computer, major parts of the Computer (Hard Ware); Input Devices, Processor, Output Devices, Storage Devices, Internal Components, Software; Types of software, System software, Application software, Computer Languages and Scripting, Booting, Computer maintenance and troubleshooting, BIOS Setting, Open Source Software and Linux OS, Navigating Linux GUI, The Internet.

7. SOFTWARE

- We have studied about the physical components or the hardware of the computer system. But the hardware is of no use on its own. Hardware needs to be operated by a set of instructions. These sets of instructions are referred to as software. It is that component of a computer system, which we cannot touch or view physically. It comprises the instructions and data to be processed using the computer hardware. The computer software and hardware complete any task together.
- **The software** comprises a set of instructions which on execution deliver the desired outcome. In other words, each software is written for some computational purpose. Some examples of software include operating systems like Ubuntu or Windows 7/11, word processing tool like LibreOffice or Microsoft Word, video player like VLC Player, photo editors and LibreOffice draw. A document or image stored on the hard disk or pen drive is referred to as a soft-copy. Once printed, the document or an image is called a hard-copy.

7.1 Need of Software

- The sole purpose of a software is to make the computer hardware useful and operational. A software knows how to make different hardware components of a computer work and communicate with each other as well as with the end-user. We cannot instruct the hardware of a computer directly. Software acts as an interface between human users and the hardware. Depending on the mode of interaction with hardware and functions to be performed, the software can be broadly classified into three categories:
  - (i) System software.
  - (ii) Programming tools.
  - (iii) Application software.

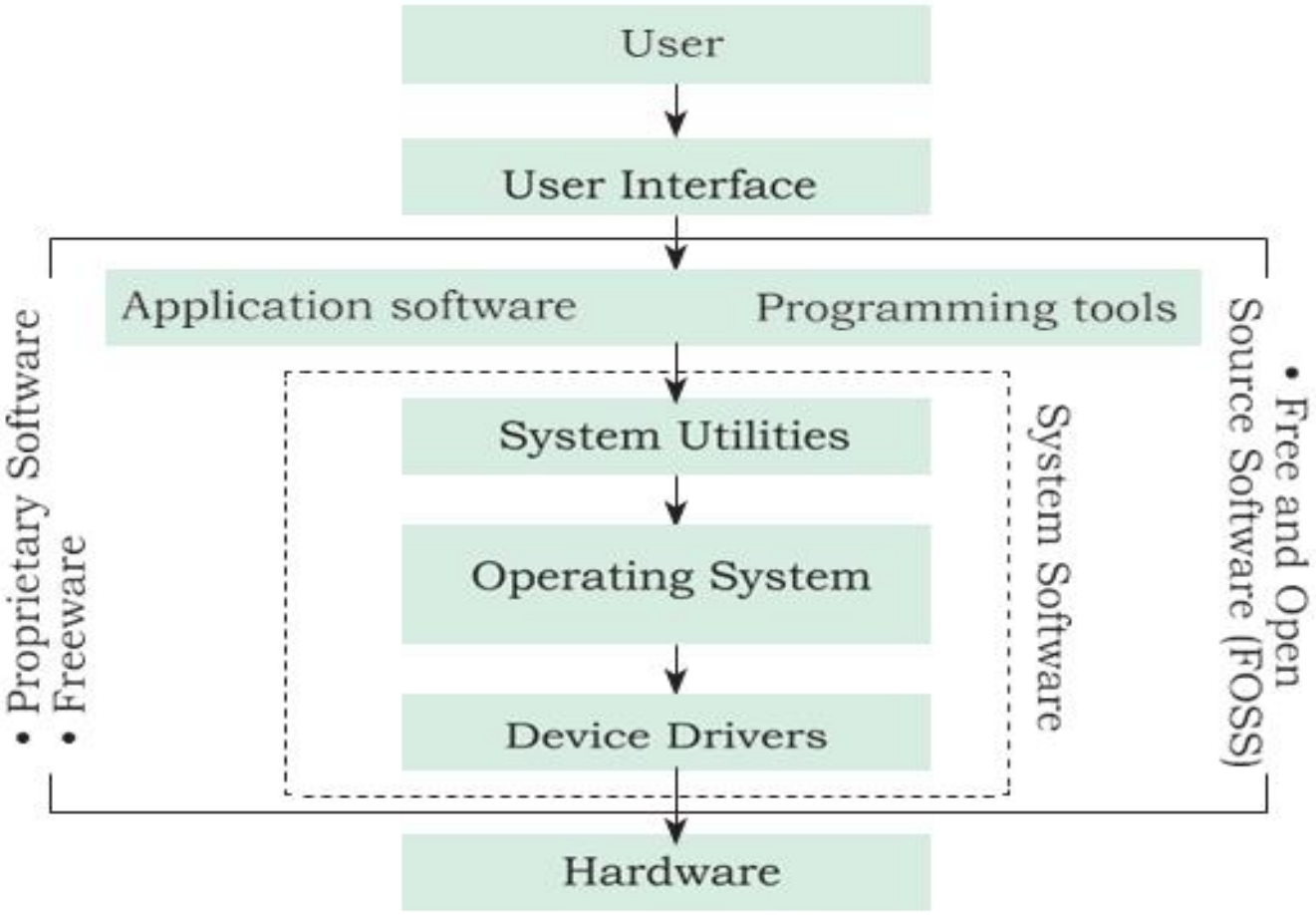
7.2 System Software

The software that provides the basic functionality to operate a computer by interacting directly with its constituent hardware is termed as system software. A system software knows how to operate and use different hardware components of a computer. It provides services directly to the end user, or to some other software. Examples of system software include operating systems, system utilities, device drivers, etc.

A. Operating System As the name implies, the operating system is a system software that operates the computer. An operating system is the most basic system software, without which other software cannot work. The operating system manages other application programs and provides access and security to the users of the system. Some of the popular operating systems are Windows, Linux, Macintosh, Ubuntu, Fedora, Android, iOS, etc.

B. System Utilities Software used for maintenance and configuration of the computer system is called system utility. Some system utilities are shipped with the operating system for example disk defragmentation tool, formatting utility, system restore utility, etc. Another set of utilities are those which are not shipped with the operating system but are required to improve the performance of the system, for example, anti-virus software, disk cleaner tool, disk compression software, etc.

C. Device Drivers As the name signifies, the purpose of a device driver is to ensure proper functioning of a particular device. When it comes to the overall working of a computer system, the operating system does the work. But everyday new devices and components are being added to a computer system. It is not possible for the operating system alone to operate all of the existing and new devices, where each device has diverse characteristics. The responsibility for overall control, operation and management of a particular device at the hardware level is delegated to its device driver. The device driver acts as an interface between the device and the operating system. It provides required services by hiding the details of operations performed at the hardware level of the device. Just like a language translator, a device driver acts as a mediator between the operating system and the attached device. The categorisation of software is shown in Figure.



Categorization of software

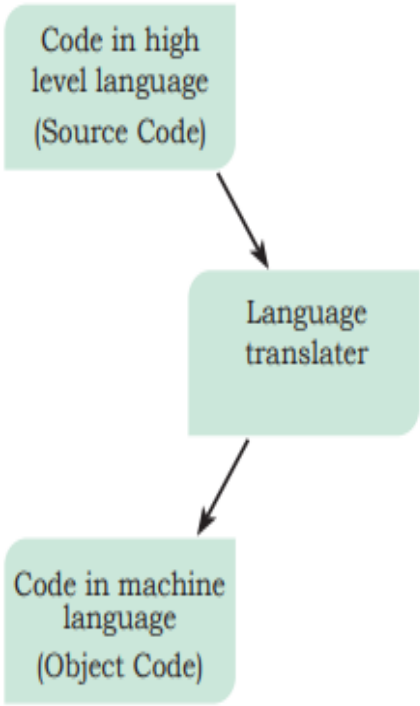
7.3 Programming Tools

- In order to get some work done by the computer, we need to give instructions which are applied on the input data to get the desired outcome. Computer languages are developed for writing these instructions. It is important to understand here that computers and humans understand completely different languages.
- While humans are able to write programs in high-level language, computers understand machine language. There is a continuous need for conversion from high level to machine level language, for which translators are needed. Also, to write the instruction, code editors (e.g., Integrated Development and Learning Environment (IDLE) in Python) are needed. We will briefly describe here the programming languages, language translators and program development tools.

A. **Classification of Programming Languages** It is very difficult for a human being to write instructions in the form of 1s and 0s. So different types of computer programming languages are developed to simplify the coding. Two major categories of computer programming languages are low-level languages and high-level languages.

- **Low-level languages** are machine dependent languages and include machine language and assembly language. Machine language uses 1's and 0's to write instructions which are directly understood and executed by the computer. But writing a code in machine language is difficult as one has to remember all operation codes and machine addresses. Also finding errors in the code written in machine language is difficult. To simplify the writing of code, assembly language was developed that allowed usage of English-like words and symbols instead of 1s and 0s. But one major drawback of writing a code in this language is that the code is computer specific, i.e., the code written for one type of CPU cannot be used for another type of CPU.
- **High level languages** are machine independent and are simpler to write code into. Instructions are using English like sentences and each high level language follows a set of rules, similar to natural languages. However, these languages are not directly understood by the computer. Hence, translators are needed to translate high-level language codes into machine language. Examples of high level language include C++, Java, Python, etc.

B. Language Translators As the computer can understand only machine language, a translator is needed to convert program written in assembly or high level language to machine language. The program code written in assembly or high-level language is called source code. The source code is converted by a translator into the machine understandable form called object (machine) code as depicted in Figure. As we have different types of computer languages, different translators are needed to convert the source code to machine code. The three types of translators used in computing systems are assembler, compiler and interpreter. The translator used to convert the code written in assembly language to machine language is called assembler. Each assembler can understand a specific microprocessor instruction set only and hence, the machine code is not portable. We also need translators to convert codes written in high level language (source code) to machine understandable form (machine code) for execution by the computer. Compiler converts the source code into machine code. If the code follows all syntactic rules of the language, then it is executed by the computer. Once translated, the compiler is not needed. An interpreter translates one line at a time instead of the whole program at one go. Interpreter takes one line, converts it into executable code if the line is syntactically correct, and then it repeats these steps for all lines in the source code. Hence, interpreter is always needed whenever a source code is to be executed.



C. Program Development Tools Whenever we decide to write a program, we need a text editor. An editor is a software that allows us to create a text file where we type instructions and store the file as the source code. Then an appropriate translator is used to get the object code for execution. In order to simplify the program development, there are software called Integrated Development Environment (IDE) consisting of text editor, building tools and debugger. A program can be typed, compiled and debugged from the IDE directly. Besides Python IDLE, Netbeans, Eclipse, Atom, Lazarus are few other examples of IDEs. Debugger, as the name implies, is the software to detect and correct errors in the source code.

7.4 Application Software

The system software provides the core functionality of the computer system. However, different users need the computer system for different purposes depending upon their requirements. Hence, a new category of software is needed to cater to different requirements of the endusers. This specific software that works on top of the system software is termed as application software. There are again two broad categories of application software — general purpose and customised application software.

A. General Purpose Software The application software developed for generic applications, to cater to a bigger audience in general are called general purpose software. Such ready-made application software can be used by end users as per their requirements. For example, spreadsheet tool Calc of LibreOffice can be used by any computer user to do calculation or to create account sheet. Adobe Photoshop, GIMP, Mozilla web browser, iTunes, etc., fall in the category of general purpose software.

B. Customised Software These are custom or tailor-made application software, that are developed to meet the requirements of a specific organisation or an individual. They are better suited to the needs of an individual or an organisation, considering that they are designed as per special requirements. Some examples of user-defined software include websites, school management software, accounting software, etc. It is similar to buying a piece of cloth and getting a tailor-made garment with the fitting, colour, and fabric of our choice

7.5 Proprietary or Free and Open Source Software

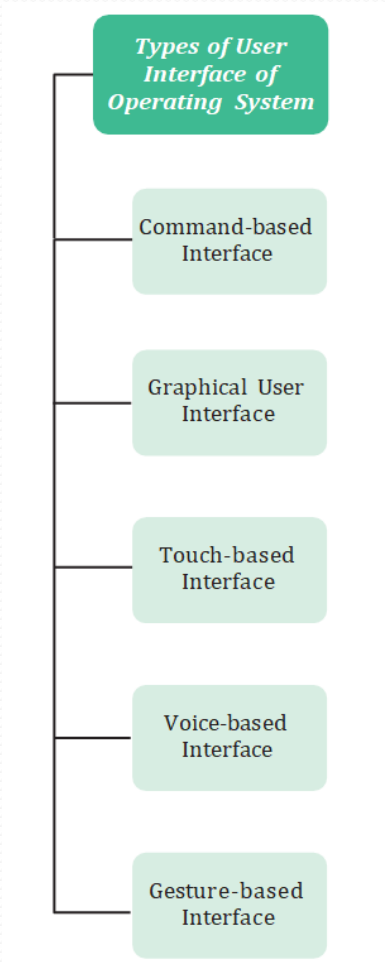
The developers of some application software provide their source code as well as the software freely to the public, with an aim to develop and improve further with each other's help. Such software is known as Free and Open Source Software (FOSS). For example, the source code of operating system Ubuntu is freely accessible for anyone with the required knowledge to improve or add new functionality. More examples of FOSS include Python, Libreoffice, Openoffice, Mozilla Firefox, etc. Sometimes, software are freely available for use but source code may not be available. Such software are called freeware. Examples of freeware are Skype, Adobe Reader, etc. When the software to be used has to be purchased from the vendor who has the copyright of the software, then it is a proprietary software. Examples of proprietary software include Microsoft Windows, Tally, Quickheal, etc. A software can be freeware or open source or proprietary software depending upon the terms and conditions of the person or group who has developed and released that software.

8. Operating System

- An operating system (OS) can be considered to be a resource manager which manages all the resources of a computer, i.e., its hardware including CPU, RAM, Disk, Network and other input-output devices. It also controls various application software and device drivers, manages system security and handles access by different users. It is the most important system software. Examples of popular OS are Windows, Linux, Android, Macintosh and so on. The primary objectives of an operating system are two-fold. The first is to provide services for building and running application programs. When an application program needs to be run, it is the operating system which loads that program into memory and allocates it to the CPU for execution. When multiple application programs need to be run, the operating system decides the order of the execution. The second objective of an operating system is to provide an interface to the user through which the user can interact with the computer. A user interface is a software component which is a part of the operating system and whose job is to take commands or inputs from a user for the operating system to process.

8.1 OS User Interface

There are different types of user interfaces each of which provides a different functionality. Some commonly used interfaces are shown in Figure.



A. Command-based Interface. Command-based interface requires a user to enter the commands to perform different tasks like creating, opening, editing or deleting a file, etc. The user has to remember the names of all such programs or specific commands which the operating system supports. The primary input device used by the user for command based interface is the keyboard. Command based interface is often less interactive and usually allows a user to run a single program at a time. Examples of operating systems with command-based interface include MS-DOS and Unix.

B. Graphical User Interface. Graphical User Interface (GUI) lets users run programs or give instructions to the computer in the form of icons, menus and other visual options. Icons usually represent files and programs stored on the computer and windows represent running programs that the user has launched through the operating system. The input devices used to interact with the GUI commonly include the mouse and the keyboard. Examples of operating systems with GUI interfaces include Microsoft Windows, Ubuntu, Fedora and Macintosh, among others.

C. Touch-based Interface. Today smartphones, tablets and PCs allow users to interact with the system simply using the touch input. Using the touchscreen, a user provides inputs to the operating system, which are interpreted by the OS as commands like opening an app, closing an app, dialing a number, scrolling across apps, etc. Examples of popular operating systems with touchbased interfaces are Android and iOS. Windows 8.1 and 10 also support touch-based interfaces on touchscreen devices.

D. Voice-based Interface. Modern computers have been designed to address the needs of all types of users including people with special needs and people who want to interact with computers or smartphones while doing some other task. For users who cannot use the input devices like the mouse, keyboard, and touchscreens, modern operating systems provide other means of human-computer interaction. Users today can use voice-based commands to make a computer work in the desired way. Some operating systems which provide voice-based control to users include iOS (Siri), Android (Google Now or "OK Google"), Microsoft Windows 10 (Cortana) and so on.

E. Gesture-based Interface. Some smartphones based on Android and iOS as well as laptops let users interact with the devices using gestures like waving, tilting, eye motion and shaking. This technology is evolving faster and it has promising potential for application in gaming, medicine and other areas.

8.2 Functions of Operating System

Now let us explore the important services and tasks that an operating system provides for managing the computer system.

A. Process Management While a computer system is operational, different tasks are running simultaneously. A program is intended to carry out various tasks. A task in execution is known as process. We can activate a system monitor program that provides information about the processes being executed on a computer. In some systems it can be activated using Ctrl+Alt+Delete. It is the responsibility of operating system to manage these processes and get multiple tasks completed in minimum time. As CPU is the main resource of computer system, its allocation among processes is the most important service of the operating system. Hence process management concerns the management of multiple processes, allocation of required resources, and exchange of information among processes.

B. Memory Management Primary or main memory of a computer system is usually limited. The main task of memory management is to give (allocate) and take (free) memory from running processes. Since there are multiple processes running at a time, there arises a need to dynamically (on-the-go) allocate and free memory to the processes. Operating system should do it without affecting other processes that are already residing in the memory and once the process is finished, it is again the responsibility of the operating system to take the memory space back for reutilisation. Hence, memory management concerns with management of main memory so that maximum memory is occupied or utilised by large number of processes while keeping track of each and every location within the memory as free or occupied.

C. File Management Data and programs are stored as files in the secondary storage of a computer system. File management involves the creation, updation, deletion and protection of these files in the secondary memory. Protection is a crucial function of an operating system, as multiple users can access and use a computer system. There must be a mechanism in place that will stop users from accessing files that belong to some other user and have not been shared with them. File management system manages secondary memory, while memory management system handles the main memory of a computer system.

D. Device Management A computer system has many I/O devices and hardware connected to it. Operating system manages these heterogeneous devices that are interdependent. The operating system interacts with the device driver and the related software for a particular device. The operating system must also provide the options for configuring a particular device, so that it may be used by an end user or some other device. Just like files, devices also need security measures and their access to different devices must be restricted by the operating system to the authorised users, software and other hardware only.

Exercises

1. Name the software required to make a computer functional. Write down its two primary services.
2. How does the computer understand a program written in high level language?
3. Why is the execution time of the machine code less than that of source code?
4. What is the need of RAM? How does it differ from ROM?
5. What is the need for secondary memory?
6. How do different components of the computer communicate with each other?
7. Draw the block diagram of a computer system. Briefly write about the functionality of each component.
8. What is the primary role of system bus? Why is data bus is bidirectional while address bus is unidirectional?
9. Differentiate between proprietary software and freeware software. Name two software for each type.
10. Write the main difference between microcontroller NOTES and microprocessor. Why do smart home appliances have a microcontroller instead of microprocessor embedded in them?
11. Mention the different types of data that you deal with while browsing the Internet.
12. Categorise the following data as structured, semi structured and unstructured:
    - Newspaper
    - Cricket Match Score
    - HTML Page
    - Patient records in a hospital

14. Name the input or output device used to do the following:
    a) To output audio
    b) To enter textual data
    c) To make hard copy of a text file
    d) To display the data or information
    e) To enter audio-based command
    f) To build 3D models
    g) To assist a visually-impaired individual in entering data
15. Identify the category (system, application, programming tool) of the following software:
    a) Compiler
    b) Assembler
    c) Ubuntu
    d) Text editor