Informatics Practices (New Syllabus)
Unit 1: Programming and Computational Thinking

Basic computer organisation

Computer organization is concerned with how the various components of computer hardware operate and they are interconnected to implement the architectural specification. Let's discuss various components of computer organization.

Computer system: A computer system allows users to input, manipulate and store data. Computer system typically includes a cpu, Hard disk, Ram, monitor, keyboard, mouse and other optional components. All of these components also can be integrated into all-in-one units, such as laptop computers.
During the data processing stage, instruction sets, known as programs, are provided to let the system know what to do with the entered system data. Without these programs, the computer would not know how to process data that enters the system, and the data might be discarded. Known as a stored program computer, this type of computer is the most common in use today. It is very flexible, as it can process any task by loading a program from storage. Computer systems can work by themselves or access other devices that are external or connected with other computer systems.

Mobile System: Now a day's mobile system has taken over computers it can perform tasks that a computer can perform and can easily be carried in our pockets. Mobile systems are becoming powerful day by day. It's a single unit made of cpu , ram and memory storage, Battery all these things are embedded on motherboard.
Let's discuss some of the components of computer system

CPU: CPU is called brain of the computer. Its primary function is to execute programs. In addition CPU also controls operation of all other components such as Input and output devices, memory.

Alternately it's referred to as a processor, central processor, or microprocessor. There are two major parts of a CPU

1. Arithmetic logical Unit (ALU): ALU of a computer system is where actual execution of instruction takes place during the processing operation. To be more precise all calculations and comparisons are done at ALU.
2. Control unit (CU): A control unit (CU) handles all processor control signals. It directs all input and output flow, fetches code for instructions from micro programs and directs other units and models by providing control and timing signals. A CU component is considered the processor brain because it issues orders to just about everything and ensures correct instruction execution.

   Control Unit functions are as follows:

   - Controls sequential instruction execution
   - Interprets instructions
   - Guides data flow through different computer areas
   - Regulates and controls processor timing
   - Sends and receives control signals from other computer devices
   - Handles multiple tasks, such as fetching, decoding, execution handling and storing results

Memory: Memory is referred as RAM or primary memory. RAM stores programs and data currently being used by the CPU. The maximum amount of programs and data that a piece of RAM can store is measured in units called bytes. Modern PCs have many millions, even billions, of bytes of RAM, so RAM is measured in units called megabytes (MB) or gigabytes (GB). An average PC will have from 4 to 8 GB of RAM, although PCs may have more or less. Each piece of RAM is called a stick. One common type of stick found in today's PC is called a dual inline memory module (DIMM).

Hard Disk: A hard drive stores programs and data that are not currently being used by the CPU. Although RAM storage is measured in megabytes and gigabytes, a PC's hard drive stores much more data than a typical PC's RAM hundreds of gigabytes to terabytes. A terabyte is 1000 gigabytes. An average PC has one hard drive, although most PCs accept more. Special PCs that need to store large amounts of data, such as a large corporation's main file-storage computer, can contain many hard drives 8 to 16 drives in some cases.
The two most common types of hard drives seen in today's PCs are the older Parallel Advanced Technology Attachment (PATA) and the more modern Serial Advanced Technology

Attachment (SATA). PATA drives use a ribbon cable very similar to the one used by floppy drives, whereas SATA drives use a very narrow cable. Most motherboards only come with SATA connections, but if you look around, you can find one that supports PATA as well.

Input output: There are many devices which are used for input and output. Let's see them one by one Input devices are those devices which give input to the computer for example keyboard, mouse, joystick etc.Output devices are those devices that gives result to end user for example printer, monitor etc.

Battery: Battery is usually used to supply uninterrupted power supply to the computer battery is present in the form of UPS (Uninterruptible power supply). All UPS are measured in both watts (the true amount of power they supply in the event of a power outage) and in volt-amps (VA). Volt-amps is the amount of power the UPS could supply if the devices took power from the UPS in a perfect way. UPS provides perfect AC power, moving current smoothly back and forth 60 times a second. Power supplies, monitors, and other devices, however, may not take all of the power the UPS has to offer at every point as the AC power moves back and forth, resulting in inefficiencies. If your devices took all of the power the UPS offered at every point as the power moved back and forth, VA would equal watts. There are two main types of UPS: online, where devices are constantly powered through the UPS's battery, and standby, where devices connected to the UPS receive battery power only when the AC sags below ~80–90 V. Another type of UPS is called line-interactive, which is similar to a standby UPS but has special circuitry to handle moderate AC sags and surges without the need to switch to battery power.

Power: Your PC uses DC voltage, so some conversion process must take place before the PC can use AC power from the power company. The power supply in a computer converts high-voltage AC power from the wall socket to low-voltage DC. The first step in powering the PC, therefore, is to get and maintain a good supply of AC power. Second, you need a power supply to convert AC to the proper voltage and amperage of DC power for the motherboard and peripherals. Finally, you need to control the by-product of electricity use, namely heat.

Informatics Practices (New Syllabus)
Unit 1: Programming and Computational Thinking

History of Computers

One of the earliest machines designed to assist people in calculations was the **abacus** which is still being used some 5000 years after its invention.

In 1642 Blaise Pascal (a famous French mathematician) invented an adding machine based on mechanical gears in which numbers were represented by the cogs on the wheels.

Englishman, Charles Babbage, invented in the 1830's a "Difference Engine" made out of brass and pewter rods and gears, and also designed a further device which he called an "Analytical Engine". His design contained the five key characteristics of modern computers:-

1. An input device
2. Storage for numbers waiting to be processed
3. A processor or number calculator
4. A unit to control the task and the sequence of its calculations
5. An output device

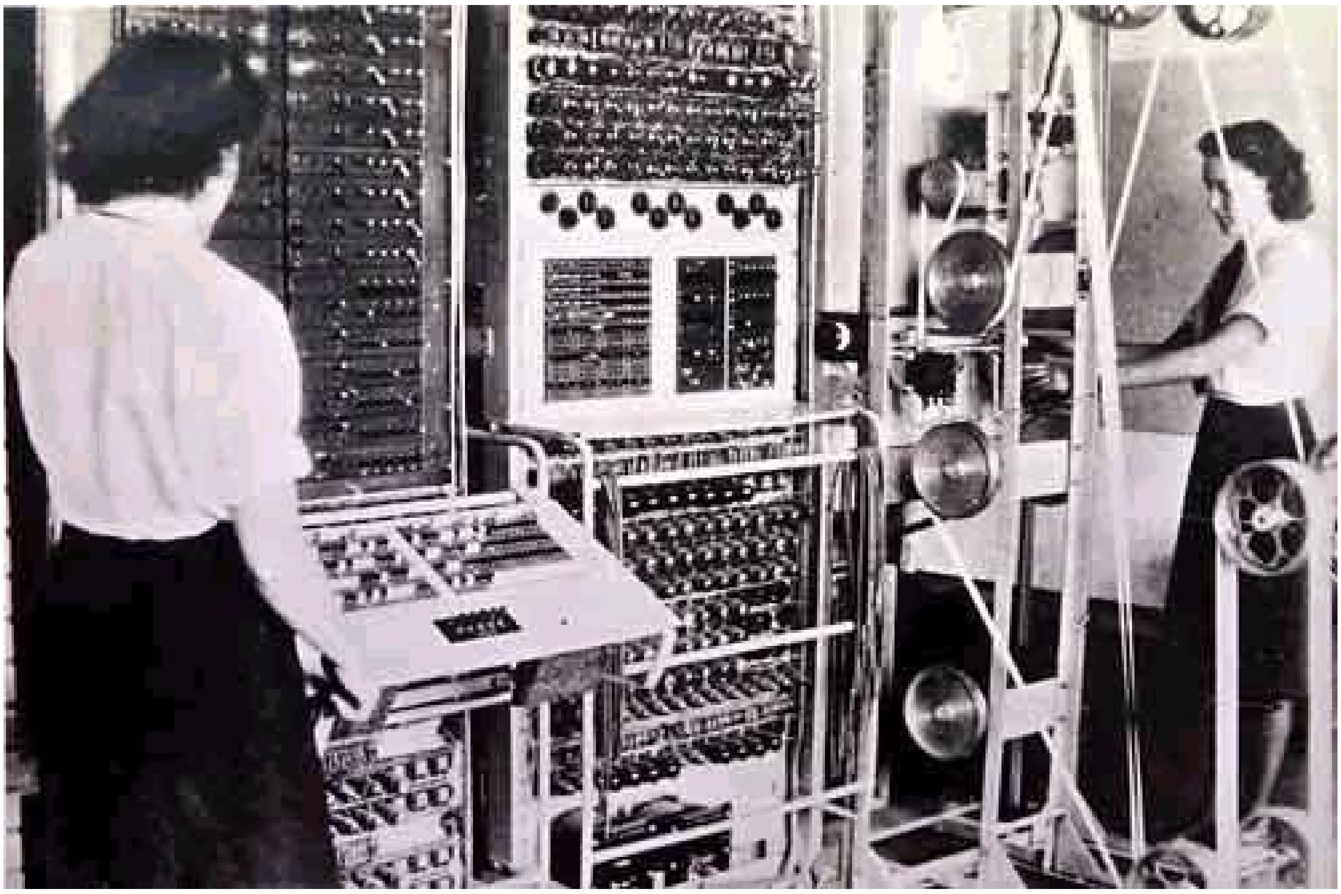
Augusta Ada Byron (later Countess of Lovelace) was an associate of Babbage who has become known as the first computer programmer.

An American, Herman Hollerith, developed (around 1890) the first electrically driven device. It utilised punched cards and metal rods which passed through the holes to close an electrical circuit and thus cause a counter to advance. This machine was able to complete the calculation of the 1890 U.S. census in 6 weeks compared with 7 1/2 years for the 1880 census which was manually counted.

In 1936 Howard Aiken of Harvard University convinced Thomas Watson of IBM to invest $1 million in the development of an electromechanical version of Babbage's analytical engine. The Harvard Mark 1 was completed in 1944 and was 8 feet high and 55 feet long.

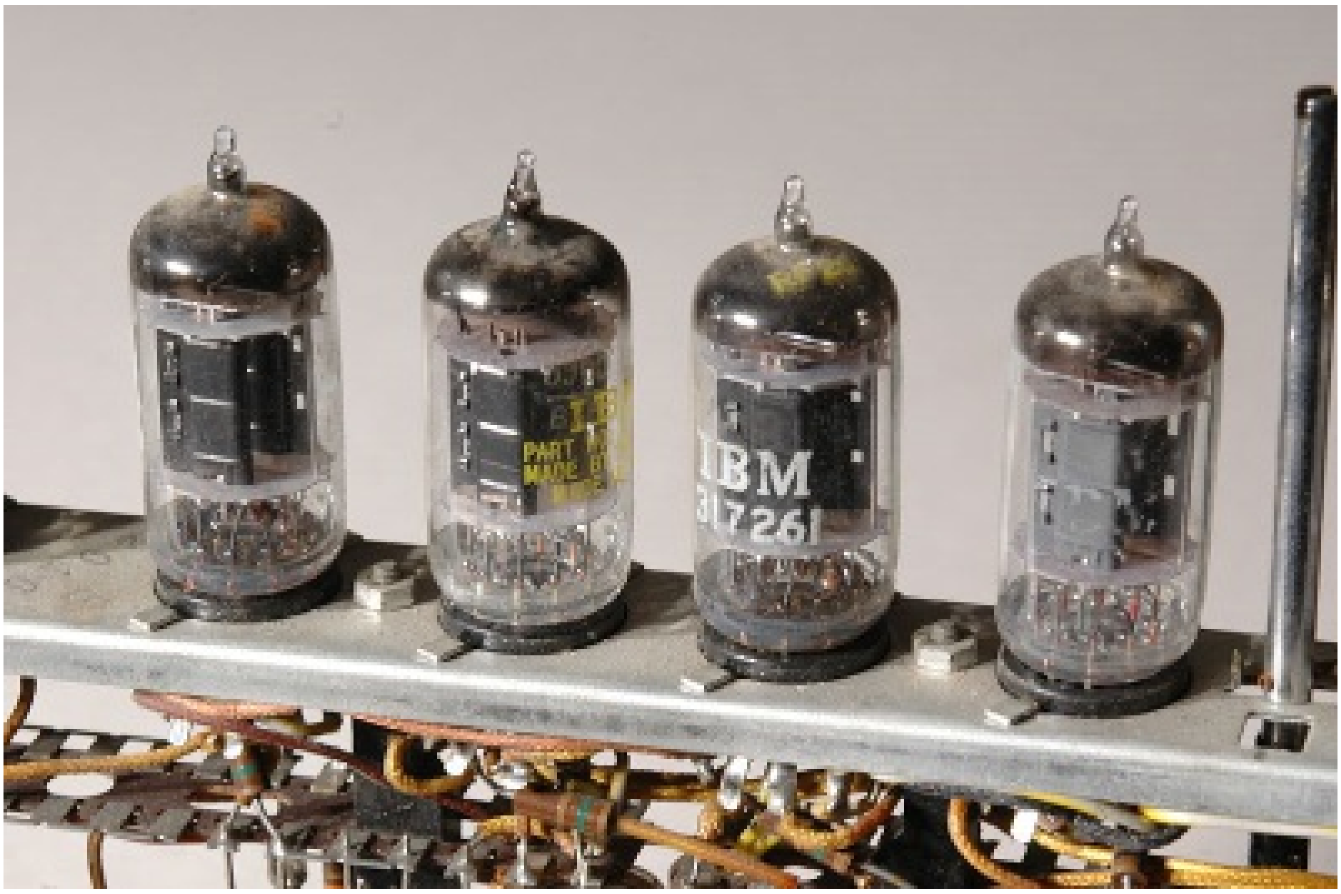
At about the same time (the late 1930's) John Atanasoff of Iowa State University and his assistant Clifford Berry built the first digital computer that worked electronically, the ABC (Atanasoff-Berry Computer). This machine was basically a small calculator.

In 1943, as part of the British war effort, a series of vacuum tube based computers (named Colossus) were developed to crack German secret codes. The Colossus Mark 2 series (pictured) consisted of 2400 vacuum tubes.


Colossus Mark 2

John Mauchly and J. Presper Eckert of the University of Pennsylvania developed these ideas further by proposing a huge machine consisting of 18,000 vacuum tubes. ENIAC (Electronic Numerical Integrator And Computer) was born in 1946. It was a huge machine with a huge power requirement and two major disadvantages. Maintenance was extremely difficult as the tubes broke down regularly and had to be replaced, and also there was a big problem with overheating. The most important limitation, however, was that every time a new task needed to be performed the machine need to be rewired. In other words programming was carried out with a soldering iron.

In the late 1940's John von Neumann (at the time a special consultant to the ENIAC team) developed the EDVAC (**E**lectronic **D**iscrete **V**ariable **A**utomatic **C**omputer) which pioneered the "stored program concept". This allowed programs to be read into the computer and so gave birth to the age of **general-purpose** computers.



Tubes from a 1950s comupter

The Generations of Computers

It used to be quite popular to refer to computers as belonging to one of several "generations" of computer. These generations are:-

The First Generation (1943-1958): This generation is often described as starting with the delivery of the first *commercial* computer to a business client. This happened in 1951 with the delivery of the UNIVAC to the US Bureau of the Census. This generation lasted until about the end of the 1950's (although some stayed in operation much longer than that). The main defining feature of the first generation of computers was that **vacuum tubes** were used as internal computer components. Vacuum tubes are generally about 5-10 centimeters in length and the large numbers of them required in computers resulted in huge and extremely expensive machines that often broke down (as tubes failed).

The Second Generation (1959-1964): In the mid-1950's Bell Labs developed the transistor. Transistors were capable of performing many of the same tasks as vacuum tubes but were only a fraction of the size. The first transistor-based computer was produced in 1959. Transistors were not only smaller, enabling computer size to be reduced, but they were faster, more reliable and consumed less electricity.

The other main improvement of this period was the development of computer languages. **Assembler languages** or **symbolic languages** allowed programmers to specify instructions in words (albeit very cryptic words) which were then translated into a form that the machines could understand (typically series of 0's and 1's: Binary code). **Higher level languages** also came into being during this period. Whereas assembler languages had a one-to-one correspondence between their symbols and actual machine functions, higher level language commands often represent complex sequences of machine codes. Two higher-level languages developed during this period (Fortran and Cobol) are still in use today though in a much more developed form.

The Third Generation (1965-1970): In 1965 the first **integrated circuit (IC)** was developed in which a complete circuit of hundreds of components were able to be placed on a single silicon chip 2 or 3 mm square. Computers using these IC's soon replaced transistor based machines. Again, one of the major advantages was size, with computers becoming more powerful and at the same time much smaller and cheaper. Computers thus became accessible to a much larger audience. An added advantage of smaller size is that electrical signals have much shorter distances to travel and so the speed of computers increased.

Another feature of this period is that computer software became much more powerful and flexible and for the first time more than one program could share the computer's resources at the same time (multi-tasking). The majority of programming languages used today are often referred to as 3GL's (3rd generation languages) even though some of them originated during the 2nd generation.

The Fourth Generation (1971-present): The boundary between the third and fourth generations is not very clear-cut at all. Most of the developments since the mid 1960's can be seen as part of a continuum of gradual miniaturisation. In 1970 **large-scale integration** was achieved where the equivalent of thousands of integrated circuits were crammed onto a single silicon chip. This development again increased computer performance (especially reliability and speed) whilst reducing computer size and cost. Around this time the first complete general-purpose **microprocessor** became available on a single chip. In 1975 **Very Large Scale Integration (VLSI)** took the process one step further. Complete computer central processors could now be built into one chip. The **microcomputer** was born. Such chips are far more powerful than ENIAC and are only about 1cm square whilst ENIAC filled a large building.

During this period **Fourth Generation Languages (4GL's)** have come into existence. Such languages are a step further removed from the computer hardware in that they use language much like natural language. Many database languages can be described as 4GL's. They are generally much easier to learn than are 3GL's.

The Fifth Generation (the future): The "fifth generation" of computers were defined by the Japanese government in 1980 when they unveiled an optimistic ten-year plan to produce the next generation of computers. This was an interesting plan for two reasons. Firstly, it is not at all really clear what the fourth generation is, or even whether the third generation had finished yet. Secondly, it was an attempt to define a generation of computers before they had come into existence. The main requirements of the 5G machines was that they incorporate the features of **Artificial Intelligence, Expert Systems, and Natural Language**. The goal was to produce machines that are capable of performing tasks in similar ways to humans, are capable of learning, and are capable of interacting with humans in natural language and preferably using both speech input (speech recognition) and speech output (speech synthesis). Such goals are obviously of interest to linguists and speech scientists as natural language and speech processing are key components of the definition. As you may have guessed, this goal has not yet been fully realised, although significant progress has been made towards various aspects of these goals.

Parallel Computing

Up until recently most computers were serial computers. Such computers had a single processor chip containing a single processor. Parallel computing is based on the idea that if more than one task can be processed simultaneously on multiple processors then a program would be able to run more rapidly than it could on a single processor. The supercomputers of the 1990s, such as the Cray computers, were extremely expensive to purchase (usually over $1,000,000) and often required cooling by liquid helium so they were also very expensive to run. Clusters of networked computers (eg. a Beowulf culster of PCs running Linux) have been, since 1994, a much cheaper solution to the problem of fast processing of complex computing tasks. By 2008, most new desktop and laptop computers contained more than one processor on a single chip (eg. the Intel "Core 2 Duo" released in 2006 or the Intel "Core 2 Quad" released in 2007). Having multiple processors does not necessarily mean that parallel computing will work automatically. The operating system must be able to distribute programs between the processors (eg. recent versions of Microsoft Windows and Mac OS X can do this). An individual program will only be able to take advantage of multiple processors if the computer language it's written in is able to distribute tasks within a program between multiple processors. For example, OpenMP supports parallel programming in Fortran and C/C++.

Basic Python Programming and Simple Data Types

Familiarization with the basics of Python programming

Guido van Rossum created the Python programming language in the late 1980s. In contrast to other popular languages such as C, C++, Java, and C#, Python strives to provide a simple but powerful syntax. Python is used for software development at companies and organizations such as Google, Yahoo and NASA.

Simple Hello program: Python programs must be written with a particular structure. The syntax must be correct, or the interpreter will generate error messages and not execute the program. When we want to write a program, we use a text editor to write the Python instructions into a file, which is called a script. By convention, Python scripts have names that end with .py. For example we will name our first program as “hello.py”.

```
>>print(“This is my first program”)
```

We will consider two ways in which we can run “hello.py”:

1. Enter the program directly into IDLE’s interactive shell
2. Enter the program into IDLE’s editor, save it, and run it.

IDLE’s interactive shell:

IDLE is a simple Python integrated development environment available for Windows, Linux, and Mac OS X. To open IDLE interactive shell “Goto start menu on windows -> open IDLE”. You may type the above one line Python program directly into IDLE and press enter to execute the program. Since it does not provide a way to save the code you enter, the interactive shell is not the best tool for writing larger programs. The IDLE interactive shell is useful for experimenting with small snippets of Python code.

IDLE’s editor: IDLE has a built in editor. From the IDLE menu, select New Window, Type the text **print("This is my first program")** into the editor. You can save your program using the Save option in the File menu. Save the code to a file named "hello.py". The extension .py is the extension used for Python source code. We can run the program from within the IDLE editor by pressing the F5 function key or from the editor's Run menu: Run -> Run Module. The output appears in the IDLE interactive shell window.

Print("This is my first program")

This is a Python statement. A statement is a command that the interpreter executes. This statement prints the message "This is my first program" on the screen. A statement is the fundamental unit of execution in a Python program. Statements may be grouped into larger chunks called blocks, and blocks can make up more complex statements.

The statement print("This is my first program") makes use of a built in function named "print". Python has a variety of different kinds of statements that may be used to build programs.

**We generally write a computer program using a high-level language. A high-level language is one which is understandable by us humans. It contains words and phrases from the English (or other) language. But a computer does not understand high-level language. It only understands program written in 0's and 1's in binary, called the machine code. A program written in high-level language is called a source code. We need to convert the source code into machine code and this is accomplished by compilers and interpreters. Hence, a compiler or an interpreter is a program that converts program written in high-level language into machine code understood by the computer.
**Interpreter: Interpreter executes one statement at a time.
**Compiler: Compiler scans the entire program and translates it as a whole into machine code.

Simple Data-Types

A data type, in programming, is a classification that specifies which type of value a variable has and what type of mathematical, relational or logical operations can be applied to it without causing an error. A string, for example, is a data type that is used to classify text and an integer is a data type used to classify whole numbers.

Some basic data types that python support are integer, float, string.

Integer (int): Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length. The number four (4) is an example of a numeric value. In mathematics, 4 is an integer value. Integers are whole numbers, which means they have no fractional parts, and they can be positive, negative, or zero. Examples of integers include 4, −19, 0, and −1005. In contrast, 4.5 is not an integer, since it is not a whole number. Python supports a number of numeric and non-numeric values. In particular, Python programs can use integer values. Let's take an example integer.py

```
X=10

Y=223313445534

Z=-987

Print(X)

Print(Y)

Print(Z)

Print(X+Y+Z)
```

In above program there are three variable holding integer values in x,y and z. Then there are three print statements which will display the three integer variables and last print statement will display addition of all the 3 numbers that are in the variables.

Float: Many computational tasks require numbers that have fractional parts. For example, to compute the area of a circle given the circle's radius, the value pi, or approximately 3.14159 is used. Python supports such non integer numbers, and they are called floating point numbers. The name implies that during mathematical calculations the decimal point can move or "float" to various positions within the number to maintain the proper number of significant digits. The Python name for the floating-point type is float. Let's take an example of float in this program we will calculate area of circle

```
PI=3.14
```

```
Radius=5.5

Area=PI*Radius*Radius

Print('Area of circle is ', Area)
```

In the above example we are taking 3 float variables PI Radius and area. In a statement Area=PI*Radius*radius we are calculating the area of circle and putting the value in variable area. Then in print statement we are printing the area of circle with a message.

String: String literals in python are surrounded by either single quotation marks, or double quotation marks. 'hello' is the same as "hello". String variable holds the text.
Let's take an example in this example we will see how to print string variables and how to take input from the user.

```
x="hello"
print("Enter text")
y=input()
print(x,y)
```

In above program there are 2 variables x and y. x is holding the string "hello" and y will take text from the keyboard at run time and next line will print both the strings.