Chapter 10:

Computer Science

Class XI (As per CBSE Board)

New Syllabus 2018-19

Visit: python.mykvs.in for regular updates
Introduction

A computer is an electronic device, under the control of instructions stored in its memory that can accept data (input), process the data according to specified rules (Program), produce information (output), and store the information for future use.

**Data vs Information**

Data are raw numbers or other findings which, by themselves, are of limited value. Information is data that has been converted into a meaningful and useful context.

Computers are being used extensively nowadays in everyday life/every field in the form of laptop, desktop, smartphone, gadgets etc.
Functionalities of a computer

Any digital computer performs five functions in gross term.

1. Take data as input
2. Stores data/instructions
3. Process those stored data
4. Generate the output
5. Control all above steps
Basic Computer Organization

Functional components of a computer

Visit: python.mykvs.in for regular updates
Basic Computer Organization

**Input/Output Units**

**Input Unit**
A device through which data and programs from the outside world enter the computer system.
Suggest – any 3 name of input devices.

**Output unit**
A device through which results stored in the computer memory are made available outside the computer system.
Suggest – any 3 name of output devices.
Basic Computer Organization

Control Unit
Control unit
It organizes the computer to work computer as single unit

Arithmetic/Logic Unit
Performs basic arithmetic operations such as addition and subtraction
Performs logical operations such as AND, OR, and NOT. Most modern ALUs have a small amount of special storage units called registers that can be accessed faster than main memory.

Memory
A collection of cells, each with a unique physical address
Most computers are byte-addressable
Cell at address 11111110 contains 10101010
# Basic Computer Organization

## Memory Units

<table>
<thead>
<tr>
<th>UNIT</th>
<th>STORAGE</th>
<th>ABBREVIATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Binary Digit, Single 1 or 0</td>
<td>B</td>
</tr>
<tr>
<td>Nibble</td>
<td>4 bits</td>
<td>-</td>
</tr>
<tr>
<td>Byte/Octet</td>
<td>8 bits</td>
<td>B</td>
</tr>
<tr>
<td>Kilobyte</td>
<td>1024 bytes</td>
<td>KB</td>
</tr>
<tr>
<td>Megabyte</td>
<td>1024 KB</td>
<td>MB</td>
</tr>
<tr>
<td>Gigabyte</td>
<td>1024 MB</td>
<td>GB</td>
</tr>
<tr>
<td>Terabyte</td>
<td>1024 GB</td>
<td>TB</td>
</tr>
<tr>
<td>Petabyte</td>
<td>1024 TB</td>
<td>PB</td>
</tr>
<tr>
<td>Exabyte</td>
<td>1024 PB</td>
<td>EB</td>
</tr>
<tr>
<td>Zettabyte</td>
<td>1024 EB</td>
<td>ZB</td>
</tr>
<tr>
<td>Yottabyte</td>
<td>1024 ZB</td>
<td>YB</td>
</tr>
</tbody>
</table>

Visit : python.mykvs.in for regular updates
Basic Computer Organization

RAM and ROM
Random Access Memory (RAM)
Memory in which each location can be accessed and changed
Read Only Memory (ROM)
Memory in which each location can be accessed but not changed
RAM is volatile, ROM is not

Secondary Storage Devices
Magnetic Tape
mass auxiliary storage device
Secondary Storage Devices

Hard disk
Fixed Head HDD / Movable head HDD

A hard disk is a set of stacked disks. Each disk has data recorded electromagnetically in concentric circles, or tracks, on the disk.

Hard Drive Types
1. Parallel Advanced Technology Attachment (PATA)
2. Serial ATA (SATA)
3. Small Computer System Interface (SCSI)
4. Solid State Drives (SSD)

Upto 12 TB sized HDD is available in the market
Mobile System

A **Mobile Phone** is essentially a two-way radio, consisting of a radio transmitter and a radio receiver.
Mobile System

Mobile Phone Components
1. A circuit board as brains of the phone
2. An antenna
3. A liquid crystal display (LCD)
4. A keyboard / A touch screen
5. A microphone
6. A speaker
7. A battery
Different types of touchscreen

1. **TFT** (Thin Film Transistor) LCD display is used for better image quality and high resolution. Since they are cheap to manufacture, they are found in budget phones usually.

2. **IPS** (In-Place Switching) LCDs are somewhat the advanced version of TFT LCDs in a way that they offer improved displays and are more battery friendly. Hence, they are found in high end phones.

3. **RESISTIVE AND CAPACITIVE**
   There are generally two types of touchscreen LCD displays; Resistive and Capacitive. Resistive touchscreen has two layers of conductive material with a small gap between them while capacitive touchscreen consists of a layer of glass coated with transparent conductor. Capacitive screens tend to be more responsive than resistive screens and are therefore found in high end phones mostly.

4. **OLED** (Organic Light Emitting Diode) is a newer technology used in mobiles and monitors for display. They are better than LCDs because they offer fast response times, wider viewing angles and higher brightness. AMOLED (Active-Matrix Organic Light-Emitting Diode) and SUPER AMOLED displays are types of OLED display. OLED types include passive-matrix OLEDs, active-matrix LEDs and transparent OLEDs.
Mobile System

Smartphone Batteries and Their Types

*Battery plays a huge role in any smartphone*

**Lithium Polymer batteries** are the most advanced batteries available in the market right now. They are made up of plastic instead of metal, which makes them usable on a smartphone of any type. The Lithium Polymer batteries do not suffer from memory effect and offer 40 percent more battery life than others.

**Lithium ion batteries** are advanced and allow for a high charge capacity based on the size and weight of the battery. However, these batteries are slightly expensive. These lithium ion batteries will not remember the charge cycle, and as a result, the battery capacity will not be reduced.

**Nickel Cadmium** These are the cells that suffer from memory effect. And, the memory effect will result in reducing the capacity of the battery and its lifespan as well.

**Nickel Metal Hydride batteries** are kind of an upgrade to the Nickel Cadmium batteries, and they boast of the same size as the latter. Nickel Metal Hydride batteries offer 30 to 40 percent more battery juice than the others.

**Battery Size** : Measured in mAh. like 2000 mAh, 4000 mAh etc.
DEVELOPMENT OF COMPUTER

Abacus is known to be the first mechanical calculating device. Which was used to be performed addition and subtraction easily and speedily? This device was a first developed by the Egyptians in the 10th century B.C, but it was given its final shape in the 12th century A.D. by the Chinese educationists.

NAPIER’S BONES John Napier’s of Scotland invented a calculating device, in the year 1617 called the Napier Bones. In the device, Napier’s used the bone rods of the counting purpose where some no. is printed on these rods. These rods that one can do addition, subtraction, multiplication and division easily.

Pascal’s calculator In the year 1642, Blaise Pascal a French scientist invented an adding machine called Pascal’s calculator, which represents the position of digit with the help of gears in it.

Leibniz Calculator In the year 1671, a German mathematician, Gottfried Leibniz modified the Pascal calculator and he developed a machine which could perform various calculation based on multiplication and division as well.

Analytical Engine In the year 1833, a scientist from England known to be Charles Babbage invented such a machine. Which could keep our data safely? This device was the first mechanical computer. Charles Babbage is also known as the father of the computer.
### GENERATION OF COMPUTER

<table>
<thead>
<tr>
<th>Generation</th>
<th>Year</th>
<th>Characteristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>1944-59</td>
<td>Use Valves (Vacuum tubes)</td>
</tr>
<tr>
<td>2nd</td>
<td>1959-64</td>
<td>Use transistors</td>
</tr>
<tr>
<td>3rd</td>
<td>1964-75</td>
<td>Large Scale Integrated Circuits</td>
</tr>
<tr>
<td>4th</td>
<td>1975-</td>
<td>Very Large Scale Integrated Circuits</td>
</tr>
<tr>
<td>5th</td>
<td>Under development</td>
<td>“Artificial Intelligence” based computers</td>
</tr>
</tbody>
</table>
Software

It is an organized instructions/code written by programmers using any of various special computer languages for specific purpose.

Software are of two main categories:

(1) Application software: It handles specialized/common tasks a user wants to perform, such as banking, hotel management, any data processing, word processing etc.

(2) System software: controls the basic functions of a computer and hides the complexity of computer system from user and application software. E.g. Operating System, Compiler, Interpret etc.
Software

SYSTEM SOFTWARE

OPERATING SYSTEM

PROGRAMMING SOFTWARE

UTILITY SOFTWARE

APPLICATION SOFTWARE

Visit: python.mykvs.in for regular updates
(1) System software

OPERATING SYSTEM

An Operating System (OS) is a system program that controls and manages the computer resources (resource manager) so that application software can run on it.

Example: Microsoft Windows, Solaris, Linux, MAC OS, Ubuntu, Apple’s i-Phone OS etc.
HOW OPERATING SYSTEM WORKS

In any computer or mobile device, the operating system can be termed as the back bone when it comes to software. This is because it has to be there before other programs can be run. It works as a middleman (interface) between machine and user.

At the simplest level, an operating system does two things:

• It manages the hardware resources of the computer system. These resources include such things as the processor, memory, disk space, etc.

• It provides a stable, consistent way for applications to deal with the hardware without having to know all the details of the hardware.
FUNCTIONS OF OPERATING SYSTEM

• Processor management
  Loads, schedules and execute process/programs.

• Memory management
  Allocates /De-allocation of memory for program execution.

• Device management
  Communicate and controls various I/O devices.

• Storage management
  Manages and controls the storage device to provide space to program for execution & data save.

• Application interface
  API/drivers provide a way for applications to make use of hardware subsystems

• User interface
  structure for interaction between a user and the computer
Software

TYPE OF OPERATING SYSTEM

* Single-User, Single Task Operating System:
These operating systems work on single task & single user at a time. E.g. DOS

* Single-User, Multi-Task Operating System:
These operating systems works on more than one task and process them concurrently at a time. E.g. windows 95 or later version of windows

* Multiuser Operating System:
In these OS, multiple users are allowed to access the same data or information at a time via a network. E.g. Unix, Linux, Windows 7.

* Multiprocessing Operating System:
Here, a single process runs on two or more processors. All the processing and their management takes place in a parallel way, hence this OS are also called as Parallel Processing. E.g. Linux, UNIX and Windows 7.

* Embedded Operating System:
These are embedded in a device, which is located in ROM. E.g. OS of microwaves, washing machine.

* Distributed Operating System:
In these OS, the computers work in co-operation with each other.

Visit : python.mykvs.in for regular updates
PROGRAMMING SOFTWARES

Language processor/Programming Language

As the computer understand machine language (0/1) where as Humans understand High level/Human Lang.

Language Processors does the coversion task(high level to machine lang.

These are of 3 types Language processors

* Assembler
* Compiler
* Interpreter

Compilers

It convert high-level language code to machine code in one session. It takes time because it have to translate high-level code to lower-level machine language all at once and then save the executable object code to memory.

Interpreters

It translates code like a compiler but reads the code and immediately executes that code, and therefore it is initially faster than a compiler.

Assemblers

It translates an assembly language program into machine language.
A software library is a suite of programming code and data used to develop software programs and applications. It is designed to assist both the programmer for faster software development and the programming language compiler/interpret in building and executing software.
A software library is a suite of programming code and data used to develop software programs and applications. It is designed to assist both the programmer for faster software development and the programming language compiler/interpreter in building and executing software.

(2) Application software
* General Purpose application software
These are ready to use software for daily use purpose e.g. word processor, spread sheet, presentation, DBMS etc.
* Specific Purpose application software
Softwares which are designed for specific task e.g. Payroll, Hotel Mgmt, Hospital Mgmt, Stock Mgmt etc.
Language of Bits

What does a Computer Understands
Computers do not understand natural languages nor programming languages. They only understand the language of bits.

1 Bit = Binary Digit (0 or 1)
8 Bits = 1 Byte
1024 Bytes = 1 KB (Kilo Byte)
1024 KB = 1 MB (Mega Byte)
1024 MB = 1 GB (Giga Byte)
1024 GB = 1 TB (Terra Byte)
1024 TB = 1 PB (Peta Byte)
1024 PB = 1 EB (Exa Byte)
1024 EB = 1 ZB (Zetta Byte)
1024 ZB = 1 YB (Yotta Byte)
1024 YB = 1 (Bronto Byte)
1024 Brontobyte = 1 (Geop Byte)
Boolean Logic
Because of computer understands machine language (0/1) which is binary value so every operation is done with the help of these binary value by the computer.

George Boole, **Boolean logic** is a form of **algebra** in which all values are reduced to either 1 or 1.

To understand boolean logic properly we have to understand Boolean logic rule, Truth table and logic gates
Boolean Logic rules

Boolean Algebra is the mathematics we use to analyse digital gates and circuits. We can use these “Laws of Boolean” to both reduce and simplify a complex Boolean expression in an attempt to reduce the number of logic gates required.

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Boolean Algebra Law or Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>A + 1 = 1</td>
<td>Annulment</td>
</tr>
<tr>
<td>A + 0 = A</td>
<td>Identity</td>
</tr>
<tr>
<td>A . 1 = A</td>
<td>Identity</td>
</tr>
<tr>
<td>A . 0 = 0</td>
<td>Annulment</td>
</tr>
<tr>
<td>A + A = A</td>
<td>Idempotent</td>
</tr>
<tr>
<td>A . A = A</td>
<td>Idempotent</td>
</tr>
<tr>
<td>NOT A = A</td>
<td>Double Negation</td>
</tr>
<tr>
<td>A + A = 1</td>
<td>Complement</td>
</tr>
<tr>
<td>A . A = 0</td>
<td>Complement</td>
</tr>
<tr>
<td>A+B = B+A</td>
<td>Commutative</td>
</tr>
<tr>
<td>A.B = B.A</td>
<td>Commutative</td>
</tr>
<tr>
<td>A+B = A.B</td>
<td>de Morgan’s Theorem</td>
</tr>
<tr>
<td>A.B = A+B</td>
<td>de Morgan’s Theorem</td>
</tr>
</tbody>
</table>

Visit : python.mykvs.in for regular updates
Language of Bits

Boolean Expression

A Boolean expression is a logical statement that is either TRUE or FALSE.

A Boolean expression can consist of Boolean data, such as the following:

* BOOLEAN values (YES and NO, and their synonyms, ON and OFF, and TRUE and FALSE)

* BOOLEAN variables or formulas

* Functions that yield BOOLEAN results

- BOOLEAN values calculated by comparison operators.

E.g.
1. \( F(x, y, z) = x' y' z' + x y' z + x y z' + x y z \)
2. \( F'(x, y, z) = x' y z + x' y' z + x' y z' + x y' z' \)
3. \( F(x, y, z) = (x + y + z) \cdot (x+y+z') \cdot (x+y'+z) \cdot (x'+y+z) \)
De Morgan’s Law

The complement of the union of two sets is equal to the intersection of their complements and the complement of the intersection of two sets is equal to the union of their complements. These are called De Morgan’s laws.

For any two finite sets $A$ and $B$

(i) $(A \cup B)' = A' \cap B'$ (which is a De Morgan's law of union).

OR

$(A+B)'=A'.B'$

(ii) $(A \cap B)' = A' \cup B'$ (which is a De Morgan's law of intersection).

OR

$(A \cdot B)'=A'+B'$
Proof of De Morgan’s law: \((A \cup B)' = A' \cap B'\)

Let \(P = (A \cup B)'\) and \(Q = A' \cap B'\)

Let \(x\) be an arbitrary element of \(P\) then \(x \in P \Rightarrow x \in (A \cup B)'\)
\(\Rightarrow x \notin (A \cup B)\)
\(\Rightarrow x \notin A\) and \(x \notin B\)
\(\Rightarrow x \in A'\) and \(x \in B'\)
\(\Rightarrow x \in A' \cap B'\)
\(\Rightarrow x \in Q\)
Therefore, \(P \subset Q \) .................. (i)

Again, let \(y\) be an arbitrary element of \(Q\) then \(y \in Q \Rightarrow y \in A' \cap B'\)
\(\Rightarrow y \in A'\) and \(y \in B'\)
\(\Rightarrow y \notin A\) and \(y \notin B\)
\(\Rightarrow y \notin (A \cup B)\)
\(\Rightarrow y \in (A \cup B)'\)
\(\Rightarrow y \in P\)
Therefore, \(Q \subset P \) .................. (ii)

Now combine (i) and (ii) we get; \(P = Q\) i.e. \((A \cup B)' = A' \cap B'\)
Proof of De Morgan’s law: \((A \cap B)' = A' \cup B'\)

Let \(M = (A \cap B)'\) and \(N = A' \cup B'\)

Let \(x\) be an arbitrary element of \(M\) then \(x \in M \Rightarrow x \in (A \cap B)'
\(\Rightarrow x \notin (A \cap B)\)
\(\Rightarrow x \notin A\) or \(x \notin B\)
\(\Rightarrow x \in A'\) or \(x \in B'\)
\(\Rightarrow x \in A' \cup B'\)
\(\Rightarrow x \in N\)

Therefore, \(M \subset N\) ................. (i)

Again, let \(y\) be an arbitrary element of \(N\) then \(y \in N \Rightarrow y \in A' \cup B'\)
\(\Rightarrow y \in A'\) or \(y \in B'\)
\(\Rightarrow y \notin A\) or \(y \notin B\)
\(\Rightarrow y \notin (A \cap B)\)
\(\Rightarrow y \in (A \cap B)'\)
\(\Rightarrow y \in M\)

Therefore, \(N \subset M\) ................. (ii)

Now combine (i) and (ii) we get; \(M = N\) i.e. \((A \cap B)' = A' \cup B'\)
Truth table

A **truth table** is a mathematical **table** used in logic.

e.g.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(A and B)</th>
<th>(A or B)</th>
<th>not(A and B)</th>
<th>not(A or B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
**Logic Gates**

Logic gate is an idealized or physical device implementing a Boolean function. These are used to construct logic circuit.

<table>
<thead>
<tr>
<th>LOGIC GATE SYMBOL</th>
<th>DESCRIPTION</th>
<th>BOOLEAN OPER.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="AND.png" alt="AND gate symbol" /></td>
<td>The AND gate output is at logic 1 when, and only when all its inputs are at logic 1, otherwise the output is at logic 0.</td>
<td>$X = A \cdot B$</td>
</tr>
<tr>
<td><img src="OR.png" alt="OR gate symbol" /></td>
<td>The OR gate output is at logic 1 when one or more of its inputs are at logic 1. If all the inputs are at logic 0, the output is at logic 0.</td>
<td>$X = A \lor B$</td>
</tr>
<tr>
<td><img src="NAND.png" alt="NAND gate symbol" /></td>
<td>The NAND Gate output is at logic 0 when, and only when all its inputs are at logic 1, otherwise the output is at logic 1.</td>
<td>$X = \overline{A \cdot B}$</td>
</tr>
<tr>
<td><img src="NOR.png" alt="NOR gate symbol" /></td>
<td>The NOR gate output is at logic 0 when one or more of its inputs are at logic 1. If all the inputs are at logic 0, the output is at logic 1.</td>
<td>$X = \overline{A \lor B}$</td>
</tr>
<tr>
<td><img src="XOR.png" alt="XOR gate symbol" /></td>
<td>The XOR gate output is at logic 1 when one and ONLY ONE of its inputs is at logic 1. Otherwise the output is logic 0.</td>
<td>$X = A \oplus B$</td>
</tr>
<tr>
<td><img src="XNOR.png" alt="XNOR gate symbol" /></td>
<td>The XNOR gate output is at logic 0 when one and ONLY ONE of its inputs is at logic 1. Otherwise the output is logic 1. (It is similar to the XOR gate, but its output is inverted).</td>
<td>$X = A \oplus B$</td>
</tr>
<tr>
<td><img src="NOT.png" alt="NOT gate symbol" /></td>
<td>The NOT gate output is at logic 0 when its only input is at logic 1, and at logic 1 when its only input is at logic 0. For this reason it is often called an INVERTER.</td>
<td>$X = \overline{A}$</td>
</tr>
</tbody>
</table>
Logic circuit
Construct a truth tables for following circuits of logic gates

Construct the logic circuit of following
1. $C + BC$
2. $AB + BC(B + C)$