January : 2016 (CBCS)

Note : **Max. marks : 60**

(i) Attempt any five questions.

(ii) All questions carry equal marks.

Q.1 (a) Define Software. What is the difference between the system software and application software?

Ans. **Computer software :** Computer software is the set of programs that makes the hardware perform a set of tasks in particular order. Hardware and software are complimentary to each other. Both have to work together to produce meaningful results. Computer software is classified into two broad categories, system software and application software.

1. System software : System software consists of a group of programs that control the operations of a computer equipment including functions like managing memory, managing peripherals, loading, storing, and is an interface between the application programs and the computer. MS DOS (Microsoft's disk operating system), UNIX are examples of system software.

The system softwares are divided into 3 types. They are :

(i) Operating system software

(ii) Language translator

(iii) Utility software

(i) Operating system software (OS) : An **operating system** is a program designed to run other programs on a computer. The various types of operating system are :

- General purpose operating system
- Special purpose operating system
- Batch processing operating system
- Multi-user operating system
- Multi-programming operating system
- Real time operating system
- Embedded operating system

(ii) Language translator : It is another system software which convert the high level language to machine level language for the purpose of machine understanding. There are 3 types of language translator, they are as follows :

- **Compiler :** A compiler is a program that translates a source program written in high-level programming language into machine code.
- **Interpreter :** An interpreter is a program that can analyze and execute a program line by line.
- **Assembler :** An assembler is a program for converting instructions written in low-level symbolic code into machine code.

(iii) Utility software : **Utility software** is a kind of system software designed to help, analyze, configure, optimize and maintain the computer. A single piece of utility software is usually called a utility or tool. Some of the utility software are :

- Antivirus
- Memory tester

2. Application software : Software that can perform a specific task for the user, such as word processing, accounting, budgeting or payroll, fall under the category of application software.

Word processors, spreadsheets, database management systems are all examples of general purpose application software.

Types of application software are :

(a) **Word processing software :** The main purpose of this software is to produce documents. MS-Word, Word Pad, Notepad and some other text editors are some of the examples of word processing software.

(b) **Database software :** Database is a collection of related data. The purpose of this software is to organize and manage data. The advantage of this software is that you can change the way data is stored and displayed. MS access, dBase, FoxPro, paradox, and oracle are some of the examples of database software.

(c) **Spread sheet software :** The spread sheet software is used to maintain budget, financial statements, grade sheets, and sales records. The purpose of this software is organizing numbers. It also allows the users to perform simple or complex calculations on the numbers entered in rows and columns. MS-excel is one of the examples of spreadsheet software.

(d) **Presentation software :** This software is used to display the information in the form of slide show. The three main functions of presentation software is editing that allows insertion and formatting of text, including graphics in the text and executing the slide shows. The best example for this type of application software is Microsoft PowerPoint.

(e) **Multimedia software :** Media players and real players are the examples of multimedia software. This software will allow the user to create audio and videos. The different forms of multimedia software are audio converters, players, burners, video encoders and decoders.

Difference between system software and application software :

S. No.	System software	Application software
1.	System software runs the system.	An application system runs over the system software.
2.	System software is a program that runs and controls the hardware units of the system.	Application software does not control any hardware units.
3.	System programs are written using **.dll, .exe** files for windows and **.rpm** files for linux.	Application software is developed on the basis of **.dll** and **.exe** files.
4.	System software is used for general purpose.	Application software is used for a specific purpose.
5.	System software is essential for computer.	Application software is not essential for computer.
6.	The number of system software is less as compared to application software.	The number of application software is more as compared to system software.
7.	System software includes programs such as compilers, debuggers, drivers, assemblers.	Application software includes programs such as media players, word processors, and spreadsheet programs.
8.	System software can run independently.	Application software cannot run independently.

Q.1 (b) Draw the block diagram of a computer. Explain each component.

Ans. **Central processing unit (CPU) :** The CPU is the central processing unit of the computer. CPU is also referred as a processor, central processor, or microprocessor. A computer's CPU handles all instructions, it receives from hardware and software running on the computer. The CPU was first developed at Intel with the help of **Ted Hoff** and others in the early 1970's. The first processor released by Intel was the 4004 processor. CPU consists of the following features :

1. CPU is considered as the brain of the computer.
2. CPU performs all types of data processing operations.
3. It stores data, intermediate results and instructions program.
4. It controls the operation of all parts of computer.

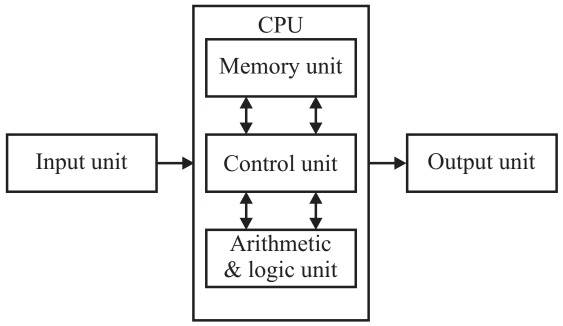


Fig. : Central processing unit (CPU)

CPU has following three components :

1. **Memory or storage unit :** This unit can store instructions, data and intermediate results. This unit supplies information to the other units of the computer when needed. It is also known as internal storage unit or main memory or primary storage or random access memory RAM. Its size affects speed, power and capability. Primary memory and secondary memory are two types of memories in the computer. All inputs and outputs are transmitted through main memory.

2. **Control unit :** This unit controls the operations of all parts of computer but does not carry out any actual data processing operations.

   Functions of control unit are :

   (i) It is responsible for controlling the transfer of data and instructions among other units of a computer.

   (ii) It manages and coordinates all the units of the computer. It obtains the instructions from the memory, interprets them, and directs the operation of the computer.

   (iii) It communicates with input/output devices for transfer of data or results from storage. It does not process or store data.

3. **Arithmetic logic unit (ALU) :** This unit consists of two subsections namely :

   (i) Arithmetic section : Function of arithmetic section is to perform arithmetic operations like addition, subtraction, multiplication and division. All complex operations are done by making repetitive use of above operations.

   (ii) Logic section : Function of logic section is to perform logic operations such as comparing, selecting, matching and merging of data.

Q.2 (a) What is Operating System? What are the different functions of an operating system?

Ans. An operating system is a program that manages the computer hardware.

1. It act as an intermediate between users of a computer and the compare hardware.
2. It controls and coordinates the use of the hardware among the various application programs for the various users.
3. An operating system (OS) is a collection of software that manages computer hardware resources and provides common services for computer programs. Application programs require an operating system to function.
   Example : Windows 10, Linux.

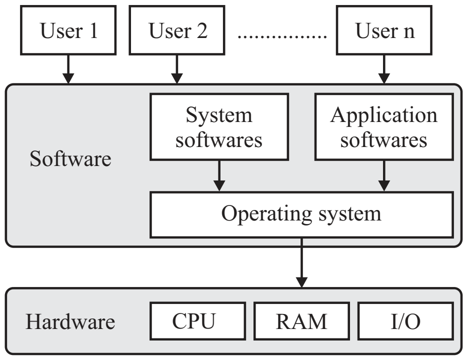


Fig. : Operating system as an interface

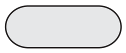
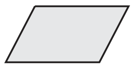
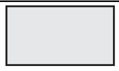
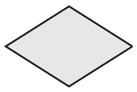
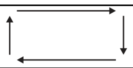
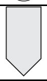
Operating system performs the following functions :

1. **Booting :** Booting is a process of starting the computer operating system to work. It checks the computer and makes it ready to work.
2. **Memory management :** It is also an important function of operating system. The memory cannot be managed without operating system. If there is no operating system, the programs may mix with each other. The system will not work properly.
3. **Loading and execution :** A program is loaded in the memory before it can be executed. Operating system provides the facility to load programs in memory easily and then execute it.
4. **Data security :** Data is an important part of computer system. The operating system protects the data stored on the computer from illegal use, modification or deletion.
5. **Disk management :** Operating system manages the disk space. It manages the stored files and folders in a proper way.
6. **Process management :** CPU can perform one task at one time. If there are many tasks, operating system decides which task should get the CPU.
7. **Device controlling :** Operating system also controls all devices attached to computer. The hardware devices are controlled with the help of small software called device drivers.
8. **Printing controlling :** Operating system also controls printing function. If a user issues two print commands at a time, it does not mix data of these files and prints them separately.
9. **Providing interface :** It is used in order that user interface acts with a computer mutually.
   The operating system offers two types of the interface to the user :
   (i) **Graphical-line interface :** It interacts with the visual environment to communicate with the computer. It uses windows, icons, menus and other graphical objects to issues commands.
   (ii) **Command-line interface :** It provides an interface to communicate with the computer by typing commands.

Q.2 (b) What is Flowchart? Explain different symbols used in Flowcharts.

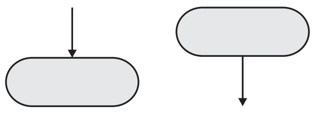
Ans. **Flowchart :** A flowchart is a pictorial (diagrammatic) representation of a program that shows the sequence of operations to be performed to arrive at the solution to a given problem. Flowcharts are quite helpful in understanding the logic of complicated problems.

A flowchart is drawn before writing an actual program to understand the sequence of actions taken by the solution to solve the problem. Flowcharts are usually drawn using some standard symbols as showed below.

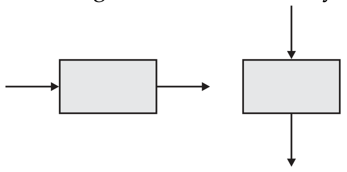
Symbol	Name	Meaning
	Terminal	Used to represent the beginning (START) or the END of a task.
	Input/Output	Used for input (reading) and output (printing) operations. The data to be read or printed are written inside the symbol.
	Processing	Used for arithmetic and data manipulation operations. The instructions are listed inside the symbol.
	Decision	Used to indicate a point at which a decision has to be made, and a branch to one of two or more alternatives is possible. The decision symbol has one entry and two exit paths. The path chosen depends on whether the answer to a question is 'yes' or 'no'.
	Flowlines	Used to indicate the flow of operation.
	Connector	Used to join different flow lines.
	Off-page connector	Used to indicate that the flowchart continues to a second page.

The following are some guidelines to draw a proper flowchart :

1. All necessary requirements should be listed out logically in order to draw an appropriate flowchart.
2. There should be no ambiguity in understanding the flowchart, i.e., the flowchart should be clear, neat and easy to follow.
3. Every flowchart must have a logical beginning and end.
4. The normal direction of the flow of a flowchart is from top to bottom or left to right.

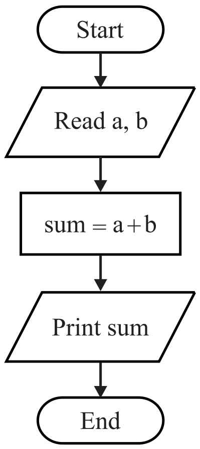


5. The number of flow line used along with the terminal symbol must be one.



6. From a process symbol, only one flow line should come out.
7. For a decision symbol only one flow line should enter, but in the case of leaving of the flow lines this number can be two to more, one for each possible answer.
8. Whenever a flowchart becomes complex it is better to use connector symbols as a substitute for flow lines.

Example : Flowchart to find the sum of two given numbers.



Advantages of flowchart :

1. **Communication :** As flowchart is a pictorial representation of a program, it is easier to explain the logic of a program to some other person by using flowcharts.
2. **Proper documentation :** Flowcharts often provide valuable documentation.
3. **Proper debugging :** Flowcharts are very efficient in detecting, locating, and removing program bugs (mistakes).

Limitations of flowchart :

1. **Laborious for complex problem :** Flowchart drawing for complex problem is very time-consuming and laborious process.
2. **Tedious task in modification :** When any change or modification is required in program logic, the flowchart may need to be redrawn completely.

Q.3 (a) Explain the significance of array. What are the different types of arrays?

Ans. **Array :** C++ programming language provides a data structure called the array, which can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type. Instead of declaring individual variables, such as number0, number1, ..., and number99, we declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables. A specific element in an array is accessed by an index. All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

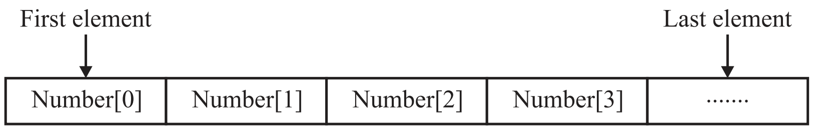


Fig. : Initializes an array

Declaring arrays :

To declare an array in C++, a programmer specifies the type of the elements and the number of elements required by an array as follows :

type array name [array size];

This is called a **single-dimensional array**. The array size must be an integer constant greater than zero and **type** can be any valid C++ data type. For example, to declare a 10-element array called **balance** of type **double**, use this statement :

double balance[10];

Now balance is a variable array which is sufficient to hold up to 10 double numbers.

Initializing arrays :

We can initialize array in C++ either one by one or using a single statement as follows :

```
double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

The number of values between braces { } cannot be larger than the number of elements that we declare for the array between square brackets [ ]. If we omit the size of the array, an array just big enough to hold the initialization is created. Therefore, if we write :

```
double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

We will create exactly the same array as we did in the previous example. Following is an example to assign a single element of the array :

```
balance[4] = 50.0;
```

The above statement assigns element number 5th in the array with a value of 50.0. All arrays have 0 as the index of their first element which is also called base index and last index of an array will be total size of the array minus 1. Following is the pictorial representation of the same array :

	0	1	2	3	4
balance	1000.0	2.0	3.4	17.0	50.0

Accessing array elements :

An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array.

Example :

```
double salary = balance[9];
```

The above statement will take 10th element from the array and assign the value to salary variable. Following is an example which will use all the above mentioned three concepts viz. declaration, assignment and accessing arrays:

```
#include <iostream>
using namespace std;
int main ()
{
        int n[ 10 ];                    // n is an array of 10 integers
        int i, j;
        for ( i = 0; i < 10; i++ )      // initialize elements of array n to 0
        {
                n[ i ] = i + 100;       // set element at location i to i + 100
        }
        for (j = 0; j < 10; j++ )       // output each array element's value
        {
                cout<<"Element[“<<i<<”] = “<<j<<n[j]<<endl;
        }
        return 0;
}
```

Output :

```
Element[0] = 100
Element[1] = 101
Element[2] = 102
Element[3] = 103
Element[4] = 104
Element[5] = 105
```

Element[6] = 106

Element[7] = 107

Element[8] = 108

Element[9] = 109

Arrays can be categorized into following types :

1. One dimensional (1-D) arrays
2. Multi-dimensional arrays

1. **One dimensional (1-D) arrays :** 1-D array is used to represent and store data in a sequential form. In 1-D array, each element is represented by a single subscript. The elements are stored in consecutive memory locations. It is also called as **single dimensional array** or **linear array**.

   Example : A[1], A[2], ….., A[N].

2. **Multi-dimensional arrays :** Multidimensional array is categorized in to two category :

   (i) Two dimensional (2-D) arrays.

   (ii) Three dimensional (3-D) arrays.

   (i) **Two dimensional (2-D) arrays :** In 2-D array each element is represented by two subscripts. In a two dimensional $m \times n$ array A has m rows and n columns and contains $m \times n$ elements. It is also called **matrix array.**

   Example : A[2][3] has 2 rows and 3 columns and $2 \times 3 = 6$ elements.

   (ii) **Three dimensional (3-D) arrays :** In 3-D array each element is represented by three subscripts. Thus a three dimensional $m \times n \times p$ array A contains $m \times n \times p$ elements.

   Example : A[2][3][2] has $2 \times 3 \times 2 = 12$ elements.

Q.3 (b) What are salient features of Object Oriented Programming? Explain.

Ans. The basic concepts of object oriented programming are :

1. Objects	2. Classes	3. Data hiding	4. Data encapsulation
5. Data abstraction	6. Inheritance	7. Polymorphism	8. Dynamic binding
9. Message passing			

1. **Objects :**

   (i) Objects are the basic run time entities in an object oriented system.

   (ii) They may represent a person, a place, a bank account, a table of data or any item that the program has to handle.

   (iii) They may also represent user defined data such as vectors, time and lists.

   (iv) Programming problem is analyzed in terms of objects and the nature of communication between them.

   (v) Program objects should be chosen such that they match closely with real world objects.

   (vi) Object take up space in the memory and have an associated address like a record in Pascal or a structure in C.

   (vii) When a program is executed, the objects interact by sending messages to one another. For example, if "customer" and "account" are two objects in a program, then the customer object may send a message to the account object requesting for the bank balance.

   (viii) Each object contain data, and code to manipulate the data.

   (ix) Objects can interact without having to know details of each other's data or code.

   (x) It is sufficient to know the type of message accepted, and the type of response returned by the objects.

   Figure below shows two notations for representing an object in an object oriented approach.

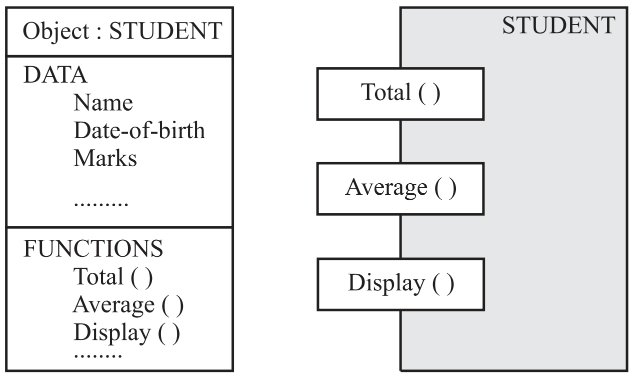


Fig. : Two ways of representing an object

2. Classes :

(i) Objects contain data, and code to manipulate that data, the entire set of data and code of an object can be made a user defined data type with the help of a class.

(ii) In fact, objects are variable of the type class.

(iii) Once a class has been defined, we can create any number of objects belonging to that class.

(iv) Each object is associated with the data of the type class with which they are created.

(v) A class is thus a collection of objects of similar type.
For example, mango, apple and orange are members of the class Fruit.

(vi) Classes are user defined data types and behaves like the built in types of a programming language.

(vii) The syntax used to create an object is no different than the syntax used to create an integer object in C. If Fruit has been defined as a class, then the statement
Fruit mango;
will create an object mango belonging to the class Fruit.

3. Data hiding :

The insulation of the data from direct access by the program is called data hiding or information hiding.

4. Data encapsulation :

(i) The wrapping up of data and functions into a single unit (called class) is known as encapsulation.

(ii) Data encapsulation is the most striking feature of a class.

(iii) The data is not accessible to the outside world, and only those functions which are wrapped in the class can access it. These functions provide the interface between the object's data and the program.

5. Data abstraction :

(i) Abstraction refers to the act of representing essential features without including the background details or implementations.

(ii) Classes use the concept of abstraction and are defined as a list of abstract attributes such as size, weight and cost, and functions to operate on these attributes.

(iii) They encapsulate all the essential properties of the objects that are to be created.

(iv) The attributes are sometimes called data members because they hold information. The functions that operate on these data are sometimes called methods or member functions.

(v) Since the classes use the concept of data abstraction, they are known as abstract data type (ADT).

6. Inheritance :

(i) Inheritance is the process by which objects of one class acquire the properties of objects of another class.

(ii) It supports the concept of hierarchical classification. For example, the bird 'robin' is a part of the class 'flying bird' which is again a part of the class 'bird'.

(iii) The principle behind this sort of division is that each derived class shares common characteristics with the class from which it is derived as shown in figure below.

(iv) In OOP, the concept of inheritance provides the idea of reusability. This means that we can add additional features to an existing class without modifying it. This is possible by deriving a new class from the existing one. The new class will have the combined features of both the classes.

(v) Note that each sub-class defines only those features that are unique to it. Without the use of classification, each class would have to explicitly include all of its features.

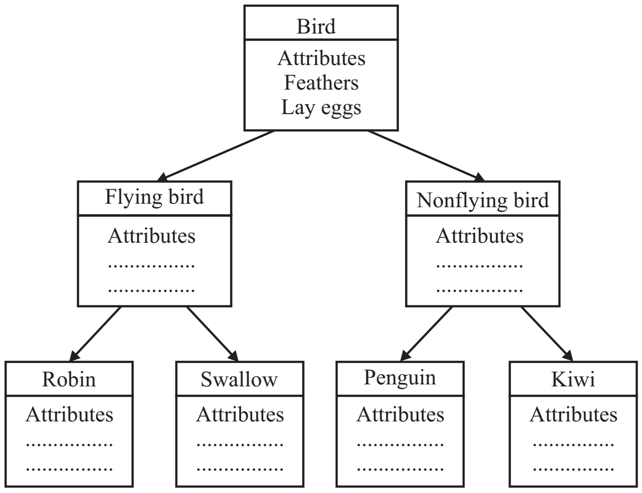


Fig. : Inheritance

7. Polymorphism :

(i) Polymorphism is another important OOP concept.

(ii) Polymorphism, Greek term, means the ability to take more than one form.

(iii) An operation may exhibit different behaviors in different instances. The behavior depends upon the types of data used in the operation.

Example : Consider the operation of addition. For two numbers, the operation will generate a sum. If the operands are strings, then the operation would produce a third string by concatenation.

(iv) The process of making an operator to exhibit different behaviors in different instances is known as **operator overloading**.

(v) Figure below shows that a single function name can be used to handle different number and different type of arguments. This is something similar to a particular word having several different meanings depending on the context. Using a single function name to perform different types of tasks is known as **function overloading**.

(vi) Polymorphism plays an important role in allowing objects having different internal structures to share the same external interface. This means that a general class of operations may be accessed in the same manner even through specific actions associated with each operation may differ. Polymorphism is extensively used in implementing inheritance.

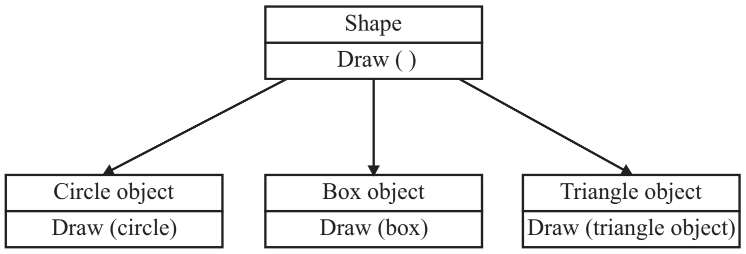


Fig. : Polymorphism

8. **Dynamic binding :**

   (i) Binding refers to the linking of a procedure call to the code to be executed in response to the call.

   (ii) Dynamic binding (also called late binding) means that the code associated with a given procedure call is not known until the time of the call at run time.

   (iii) It is associated with polymorphism and inheritance.

   (iv) A function call associated with a polymorphic reference depends on the dynamic type of that reference.

   (v) Consider the procedure "Draw" in figure above. By inheritance, every object will have this procedure. Its algorithm is, however, unique to each object and so the Draw procedure will be redefined in each class that defines the object. +At run time, the code matching the object under current reference will be called.

9. **Message passing :**

   (i) An object oriented program consists of a set of objects that communicate with each other. The process of programming in an object oriented language, therefore, involves the following basic steps :

      (a) Creating classes that define objects and their behavior,

      (b) Creating objects from class definition, and

      (c) Establishing communication among objects.

   (ii) Objects communicate with one another by sending and receiving information much the same way people pass messages to one another. The concept of message passing makes it easier to talk about building systems that directly model or simulate their real world counterparts.

   (iii) A message for an object is a request for execution of a procedure, and therefore will invoke a function (procedure) in the receiving object that generates the desired result.

   (iv) Message passing involves specifying the name of object, the name of the function (message) and the information to be sent.

   Example :

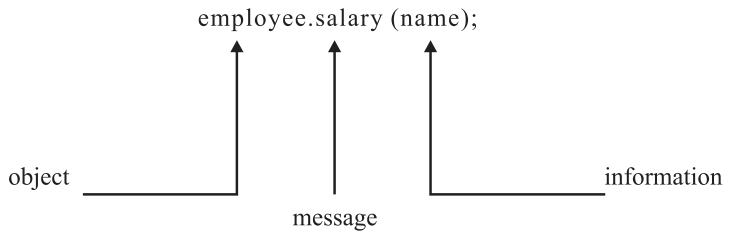


Fig. : Message passing

   (v) Objects have a life cycle. They can be created or destroyed. Communication with an object is feasible as long as it is alive.

Q.4 (a) What is class and object in C++? Explain with example.

Ans. **Class :** A class is a way to bind the data and its associated functions together. It allows the data (and functions) to be hidden, if necessary, from external use. When defining a class, we are creating a new abstract data type that can be treated like any other built-in data type.

Generally, a class specification has two parts :

1. Class declaration
2. Class function definitions

The class declaration describes the type and scope of its members. The class function definitions describe how the class functions are implemented.

The general form of a class declaration is :

```
class class_name
{
        private:
                variable declarations;
                function declarations;
        public:
                variable declarations;
                function declarations;
        protected:
                variable declarations;
                function declarations;
}
```

The class declaration is similar to a structure declaration. The keyword class specifies, that what follows is an abstract data of type class_name. The body of a class is enclosed within braces and terminated by a semicolon. The class body contains the declaration of variables and functions. These functions and variables are collectively called **class members**. They are usually grouped under two sections, namely, private and public to denote which of the members are private and which of them are public. The keywords private and public are known as **visibility labels**; these keywords are followed by a colon.

A simple class example :

A typical class declaration would look like :

```
class Item
{
                int number;                    // variables declaration
                float cost;                    // private by default
        public:
                void getdata(int a, float b);  // functions declaration
                void putdata(void);            // using prototype
};                                             // ends with semicolon
```

Creating objects :

Remember that the declaration of item as shown above does not define any objects of item but only specifies what they will contain. Once a class has been declared, we can create variables of that type by using the class name (like any other built-in type variable).

Example :

```
        item x;                                // memory for x is created
```

creates a variable x of type item. In C++, the class variables are known as **objects**. Therefore, x is called an object of type item. We may also declare more than one object in one statement.

Example :

```
        item x, y, z;
```

The declaration of an object is similar to that of a variable of any basic type. The necessary memory space is allocated to an object at this stage. Note that class specification, like a structure, provides only a template and does not create any memory space for the objects.

Objects can also be created when a class is defined by placing their names immediately after the closing brace, as we do in the case of structures. That is to say, the definition

```
        class item
        {
                ……...............
                ……...............
                ……...............
        }x, y, z;
```

would create the objects x, y and z of type item. This practice is seldom followed because we would like to declare the objects close to the place where they are used and not at the time of class definition.

Example : Consider the following program

```
#include <iostream>
using namespace std;
class item
{
                int number;                 // private by default
                float cost;                 // private by default
        public:
                void getdata(int a, float b);   // prototype declaration to be defined
                void putdata(void)          // Function defined inside class
                {
                        cout<<"number ="<<number<<"\n";
                        cout<<"cost ="<<cost<<"\n";
                }
};
                                            // Member Function Definition
void item :: getdata(int a, float b)        // use membership label
{
        number = a;                         // private variables
        cost = b;                           // directly used
}
                                            // Main Program
int main()
{
        item x;                             // create object x
        cout<<"\nobject x "<<"\n";
        x.getdata(100, 299.95);             // call member function
        x.putdata();                        // call member function
        item y;                             // create another object
        cout<<"\nobject y"<<"\n";
        y.getdata(200, 175.50);
        y.putdata();
        return 0;
}
```

Output :

```
object x
number :100
```

```
cost :299.95
object y
number :200
cost :175.5
```

Q.4 (b) What is a destructor? What is the importance of a destructor?

Ans. **Destructor :** The complement of a constructor is the destructor. This function is called when an object is destroyed. For example, an object that allocates memory when it is created and the memory is released when it is destroyed. The name of a destructor is the name of its class preceded by a tilde (~).

The destructor function has the following characteristics :

1. A destructor function never takes any arguments.
2. A destructor function will not return any values.
3. Destructors cannot be overloaded.
4. Destructors can be made virtual.
5. A class's destructor is called when an object is destroyed.
6. Local objects are destroyed when they go out of scope.
7. Global objects are destroyed when the program ends.

Example : Program to show the use of destructor.

```
#include<iostream>
using namespace std;
class Sample
{
                int a;
        public :
                Sample( )
                {
                        cout<<"\nConstructor called";
                        a = 10;
                }
                void show( )
                {
                        cout<<"\na = "<<a;
                }
                ~Sample( )
                {
                        cout<<"\nDestructor called";
                }
};
int main( )
{
        Sample s;
        s.show( );
        return 0;
}
```

Output :

```
Constructor called
a = 10
Destructor called
```

Q.5 (a) What is computer networking? Write its advantages.

Ans. **Computer network :** A computer network is a group of computer systems and other computing hardware devices that are linked together through communication channels to facilitate communication and resource sharing among a wide range of users. Networks are commonly categorized based on their characteristics.

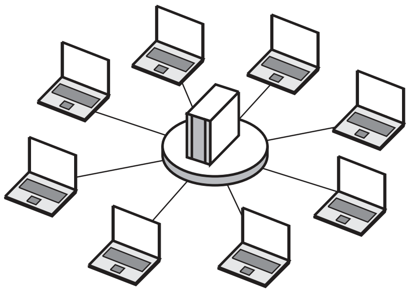


Fig. : Computer network

Advantages of computer network :

1. **Availability of information :** It allows access to a vast useful information, including traditional reference materials and facts, such as news and current affairs.
2. **File sharing easier :** Computer networking allows easier accessibility for people to share their files, which greatly helps them with saving more time and effort, since they could do file sharing more accordingly and effectively.
3. **Inexpensive system :** Installing networking software on our device would not cost too much, as we are assured that it lasts and can effectively share information to our peers. Also, there is no need to change the software regularly.
4. **Cost efficiency :** User can use a lot of software products available on the market which can just be stored or installed in your system or server, and can then be used by various workstations.
5. **Boosts storage capacity :** Since information, files and resources are shared to other people, we have to ensure all data and content are properly stored in the system.

Disadvantages of computer network :

1. **Lack of independence :** User will be dependent on the main file server, which means that, if it breaks down, the system would become useless and it make users idle.
2. **Security difficulties :** User's security would be always at risk because there would be a large number of people who would be using a computer network to get and share their files and resources.
3. **Lack of robustness :** If a computer network's main server breaks down, the entire system would become useless.
4. **Viruses and malware :** If even one computer on a network gets affected by a virus, there is a possible threat for the other systems getting affected too. Viruses can spread on a network easily, because of the inter-connectivity of workstations.
5. **Needs an efficient handler :** The technical skills and knowledge of how to operate networks and administer a computer network is considerably high. Any user with basic skills cannot do this job. Also, the accountability to manage a network is very high.

Q.5 (b) Compare the OSI and TCP/IP models.

Ans. Comparison between OSI model and TCP/IP model are :

S. No.	OSI (Open system interconnection)	TCP/IP (Transmission control protocol / Internet protocol)
1.	OSI provides layer functioning and also defines functions of all the layers.	TCP/IP model is more based on protocols and protocols are not flexible with other layers.
2.	In OSI model the transport layer guarantees the delivery of packets.	In TCP/IP model the transport layer does not guarantees delivery of packets.
3.	Follows horizontal approach.	Follows vertical approach.
4.	OSI model has a separate presentation layer.	TCP/IP does not have a separate presentation layer.
5.	OSI is a general model.	TCP/IP model cannot be used in any other application.
6.	Network layer of OSI model provide both connection oriented and connectionless service.	The network layer in TCP/IP model provides connectionless service.
7.	OSI model has a problem of fitting the protocols in the model.	TCP/IP model does not fit any protocol.
8.	Protocols are hidden in OSI model and are easily replaced as the technology changes.	In TCP/IP replacing protocol is not easy.
9.	OSI model defines services, interfaces and protocols very clearly and makes clear distinction between them.	In TCP/IP it is not clearly separated its services, interfaces and protocols.
10.	It has 7 layers.	It has 4 layers.

Q.6 (a) Define the terms data, information, knowledge and database.

Ans. **Data :** Data can be defined as a set of isolated and unrelated raw facts, represented by values, which have little or no meaning, simply because they lack a context for evaluation. Usually, the values are represented in the forms of characters, numbers, or any symbol such as 'Monica', '35', and 'chef'. Although these words and numbers have some meaning, it is difficult to figure out exactly what these values signify.

Information : It can be defined as a set of organized and validated collection of data. When the data are processed and converted into a meaningful and useful form, they are known as **information**.

Example : 'Monica is 35 years old and she is a chef'.

Knowledge : It is the act of understanding the context in which the information is used. It can be based on learning through information, experience, and/or intuition. Based on the knowledge, the information can be used in a particular context.

Example : If an hotelier uses the information about Monica (she is a chef) to hire her, he is using his knowledge. Hence, knowledge can also be referred to as a person's capability and wisdom and how much that person knows about a particular subject. Consequently, it can be said that data constitute information, and information constitutes knowledge.

Figure below shows the relationship among data, information, and knowledge.

$$\begin{tikzcd}\boxed{\text{Data}} \arrow[r, "\text{Meaningful}", "\text{Form}"'] & \boxed{\text{Information}} \arrow[r, "\text{Conclusion}", "\text{Made}"'] & \boxed{\text{Knowledge}}\end{tikzcd}$$

Fig. : Data, information and knowledge

Database : A database can be defined as a collection of related data from which users can efficiently retrieve the desired information. It can be anything from a simple collection of roll numbers, names, addresses, and phone numbers of students to a complex collection of sound, images, and even video or film clippings. Though they are generally computerized, instances of non-computerized databases from everyday life can be cited in abundance. A dictionary, a phone book, a collection of recipes, and a TV guide are examples of non-computerized databases. The examples of computerized databases include customer files, employee rosters, books catalog, equipment inventories, and sales transactions.

Difference between hierarchical, network and relational models :

S. No.	Hierarchical data model	Network data model	Relational data model
1.	Relationship between records is of parent child type.	Relationship between records is expressed in the form of pointers or links.	Relationship between records is represented by a relation that contains a key for each record involved in the relations.
2.	Many-to-many relationship cannot be expressed in this model.	Many-to-many relationship can also be implemented.	Many-to-many relationship can be easily implemented.
3.	It is a simple, straightforward and natural method of implementing record relationships.	Record relationship implementation is quite complex due to the use of pointers.	Relationship implementation is very easy though the use of a key or composite key field.
4.	This type of model is useful only when there is some hierarchical character in the database.	Network model is useful for representing such records which have many-to-many relationships.	Relational model is useful for representing most of the real world objects and relationships among them.
5.	In order to represent links among records, pointers are used. Thus relationships among records are physical.	In network model also the relationship among records are physical.	Relational model does not maintain physical connection among records. Data is organized logically in the form of rows and columns and stored in table.
6.	Searching for a record is very difficult since one can retrieve a child only after going through its parent record.	Searching a record is easy since there are multiple access paths to a data element.	A unique, indexed key field is used to search for a data element.
7.	During updating or deletion process, chance of data inconsistency is involved.	No problem of inconsistency exists in network model because a data element is physically located at just one place.	Data integrity maintaining methods like Normalization process, etc. are adopted for consistency.

Q.6 (b) What are the different kinds of data models? Compare them.

Ans. Data model : A data model describes how data are represented and accessed. Data model formally defined data elements and relationships among data elements for a domain of interest.

Types of data models :

There are basically two types of data models :

I. Record based data models.

II. Object based data models.

I. Record based data models :

In record-based models, the database is organized in fixed-format records of several types. A fixed number of fields, or attributes, are defined in each record type, and each field is usually of a fixed length.

The three most popular record-based data models are :

1. Relational data model
2. Network data model
3. Hierarchical data model

1. Relational data model :

(i) The relational model uses tables to represent the data and the relationships among those data.

(ii) Each table has multiple columns, and each column is identified by a unique name.

(iii) It is a low level model.

Name	Street	City	Number
Lowery	Maple	Queens	900
Shiver	North	Bronx	556
Shiver	North	Bronx	647
Hodges	Sidehill	Brooklyn	801
Hodges	Sidehill	Brooklyn	647

Number	Balance
900	55
556	100000
647	105366
801	10533

Fig. : Relational model

In this database, each row in the table represents a different customer. Relationships link rows from two tables on the basis of the key field, in this case number.

Advantages of relational data model :

(a) **Structural independence :** Relational database model has structural independence, i.e. changes made in the database structure do not affect the DBMS's capability to access data.

(b) **Simplicity :** The relational model is the simplest model at the conceptual level.

(c) **Ease of designing, implementation, maintenance, and usage :** The relational model makes it easy to design, implement, maintain and use the databases.

(d) **Ad-hoc query capability :** One of the main reasons for the huge popularity of the relational database model is the presence of powerful, flexible and easy to use query capability.

Disadvantages of relational data model :

(a) Hardware overheads : The RDBMS needs comparatively powerful hardware as it hides the implementation complexities and the physical data storage details from the users.

(b) Ease of design can result in bad design : As the relational database is an easy to design and use system, it can result in the development and implementation of poorly designed database management systems. As the size of the database increases, several problems may arise like system shutdown, performance degradation and data corruption.

2. Network data model :

(i) In the network model, data are represented by collections of records.

(ii) Relationships among data are represented by links.

(iii) In this model graph data structure is used.

(iv) A network model permits a record to have more than one parent.

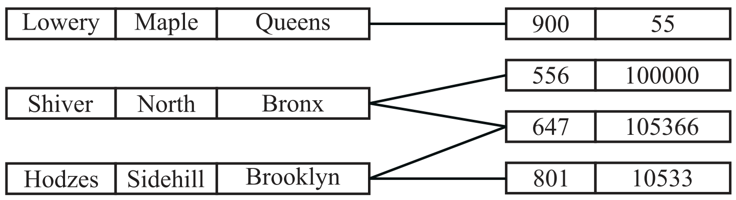


Fig. : A simple network model

Advantages of network data model :

(a) Simplicity : The network data model is also conceptually simple and easy to design.

(b) Ability to handle more relationship types : The network model can handle the one-to-many and many-to-many relationships.

(c) Ease of data access : In the network database terminology, a relationship is a set. Each set comprises of two types of records an owner record and a member record.

(d) Data integrity : In a network model, no member can exist without an owner. A user must therefore first define the owner record and then the member record. This ensures the data integrity.

(e) Data independence : The application programs work independently of the data. Any changes made in the data characteristics do not affect the application program.

(f) Database standards : The standards devised by the DBTG (Database task group of CODASYL committee) form the basis of the network model.

Disadvantages of network data model :

(a) System complexity : In a network model, data are accessed one record at a time. Therefore, a user friendly database management system cannot be created using the network model.

(b) Lack of structural independence : Making structural modifications to the database is very difficult in the network database model as the data access method is navigational. Any changes made to the database structure require the application programs to be modified before they can access data. Though the network database model achieves data independence, but it still fails to achieve structural independence.

3. Hierarchical model :

(i) In the hierarchical model, data are represented by collections of records.

(ii) Relationships among data are represented by links.

(iii) In this model tree data structure is used.

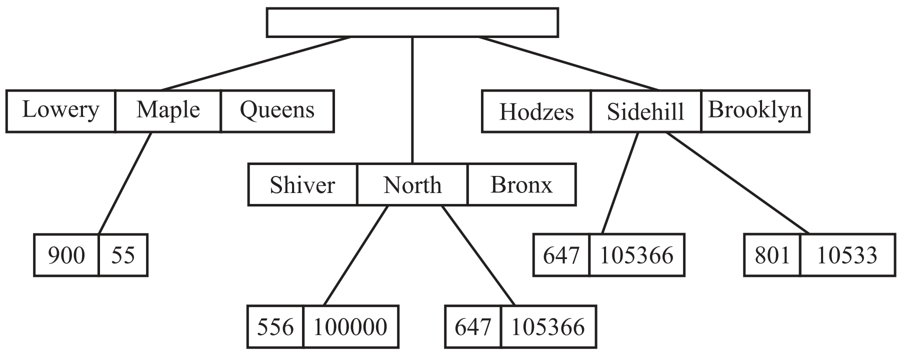


Fig. : Hierarchical model

(iv) There are two concepts associated with the hierarchical model-segment types and parent-child relationships. Segment type is similar to the record types in the network models. The information retrieved only by navigating from the root segment type to the nodes segment types. Thus we can access a segment type only via its parent segment type in the parent-child relationship. The operators provided for manipulating such structures include operators for traversing hierarchic paths up and down the trees.

Advantages of hierarchical model :

(a) **Simplicity :** Since the database is based on the hierarchical structure, the relationship between the various layers is logically simple.

(b) **Data security :** Hierarchical model was the first database that offered the data security that is provided and enforced by the DBMS.

(c) **Data integrity :** Since the hierarchical model is based on the parent/child relationship, there is always a link between the parent segment and the child segment under it. The child segments are always automatically referenced to its parent, this model promotes data integrity.

(d) **Efficiency :** The hierarchical database model is a very efficient one when the database contains a large number of one-to-many relationships and when the users require large number of transactions, using data whose relationships are fixed.

Disadvantages of hierarchical model :

(a) **Implementation complexity :** The hierarchical database model is conceptually simple and easy to design, it is quite complex to implement.

(b) **Database management problems :** If we make any changes in the database structure of a hierarchical database, then it is required to make the necessary changes in all the application programs that access the database.

(c) **Lack of structural independence :** Structural independence exists when the changes made to the database structure does not affect the DBMS's ability to access data. Hierarchical database systems use physical storage paths to navigate to the different data segments. So the application programmer should have a good knowledge of the relevant access paths to access the data. So if the physical structure is changed the applications will also have to be altered. Thus, in a hierarchical database the benefits of data independence are limited by structural dependence.

(d) **Programming complexity :** Hierarchical model requires knowledge of complex pointer systems, which is difficult for users who have little or no programming knowledge.

(e) **Implementation limitation :** The many-to-many relationships, which are more common in real life, are very difficult to implement in a hierarchical model.

II. Object based data models :

In object-based models, the database is organized in real world objects of several types. A number of fields, or attributes, are defined in each object type, and each field is usually of a variable length.

The two most popular object-based data models are :

1. Object oriented model
2. ER model

1. Object oriented model :

(i) The object-oriented model is based on a collection of objects, like the ER model.
(ii) An object contains values stored in instance variables within the object.
(iii) Unlike the record-oriented models, these values are themselves objects.
(iv) Thus objects contain objects to an arbitrarily deep level of nesting.
(v) An object also contains bodies of code that operate on the object. These bodies of code are called methods.
(vi) Objects that contain the same types of values and the same methods are grouped into classes.
(vii) A class may be viewed as a type definition for objects.
(viii) Analogy : the programming language concept of an abstract data type.
(ix) The only way in which one object can access the data of another object is by invoking the method of that other object. This is called sending a message to the object.
(x) Internal parts of the object, the instance variables and method code, are not visible externally.
(xi) Result is two levels of data abstraction.

Example : Consider an object representing a bank account.

(1) The object contains instance variables number and balance.
(2) The object contains a method pay_interest which adds interest to the balance.
(3) Under most data models, changing the interest rate entails changing code in application programs.
(4) In the object oriented model, this only entails a change within the pay_interest method.

(xii) Unlike entities in the E-R model, each object has its own unique identity, independent of the values it contains :

(1) Two objects containing the same values are distinct.
(2) Distinction is maintained in physical level by assigning distinct object identifiers.

Advantages of object oriented data model :

(a) **Capability to handle large number of different data types :** Traditional database models like hierarchical, network and relational database are limited in their capability to store the different types of data. For example, one cannot store pictures, voices and video in these databases. But the object-oriented database can store any type of data including text, numbers, pictures, voice and video.

(b) **Combination of object-oriented programming and database technology :** Perhaps the most significant characteristic of object-oriented database technology is that it combines object-oriented programming with database technology to provide an integrated application development system.

(c) **Data access :** Object-oriented database represent relationships explicitly, supporting both navigational and associative access to information.

Disadvantages of object oriented data model :

(a) **Difficult to maintain :** Object oriented data model is difficult to maintain when organizational information changes.

(b) **Not suited for all applications :** Object-oriented database systems are not suited for all applications.

2. ER model (Entity relational model) :

(i) The entity-relationship model is based on a perception of the world as consisting of a collection of basic objects (entities) and relationships among these objects.

(ii) It is an object-based logical model.

(ii) It is a high-level data model.

(iii) An entity is a distinguishable object that exists.

(iv) Each entity has associated with it a set of attributes describing it.
Example : Number and balance for an account entity.

(v) A relationship is an association among several entities.
Example : A Cust-Acct relationship associates a customer with each account he or she has.

(vi) The set of all entities or relationships of the same type is called the entity set or relationship set.

(vii) Another essential element is the ER diagram in which the mapping cardinalities express the number of entities to which another entity can be associated via a relationship set.

(viii) The overall logical structure of a database can be expressed graphically by an ER diagram :

(1) Rectangles : represent entity sets.
(2) Ellipses : represent attributes.
(3) Diamonds : represent relationships among entity sets.
(4) Lines : link attributes to entity sets and entity sets to relationships.

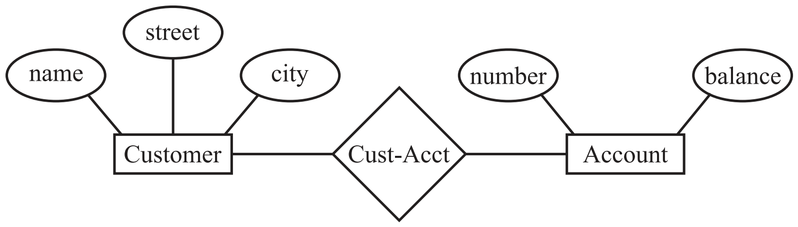


Fig. : ER model

Advantages of ER Model :

(a) Conceptually it is very simple : Making the ER diagram is a very easy process. We just know about the notations about various types of entities, their attributes and the relationships between these entities.

(b) Better visual representation : The ER model gives graphical and diagrammatical representation of various entities, their attributes and relationships between entities.

(c) Effective communication tool : It is an effective communication tool among users, domain experts and database designers.

(d) It is **highly integrated with relational model**, so converting ER models to tables is very simple.

(e) Easy conversion to any data model : Conversion of ER model to any other data model like network model, hierarchical model and the relational model is very easy.

Disadvantages of ER Model :

(a) Limited constraints and specifications. **Example :** minimum cardinality.

(b) Loss of information content. **Example :** FAN Trap, CHASM Trap.

(c) No industry standard for notation : There is no industry standard notation for developing an ER model.

(d) Popular for high level design : The ER data model is especially popular for high level design.

(e) No representation of data manipulation in ER model.

Q.7 (a) What is cloud computing? Enlist and explain characteristics of cloud computing.

Ans. **Cloud computing :** Cloud computing is a technology that uses the internet and central remote servers to maintain data and applications. Cloud computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access. This technology allows more efficient computing by centralizing data storage, processing and bandwidth. Cloud computing customers do not own the physical infrastructure; rather they rent the usage from a third party provider. They consume resources as a service and pay only for resources that they use.

Some simple examples of cloud computing are **yahoomail, gmail** and **hotmail** etc. With an internet connection we can start sending e-mails. The server and e-mail management software is all on the cloud (internet) and is totally managed by the cloud service provider **yahoo, google** etc. The consumer get the software only and enjoy the benefits.

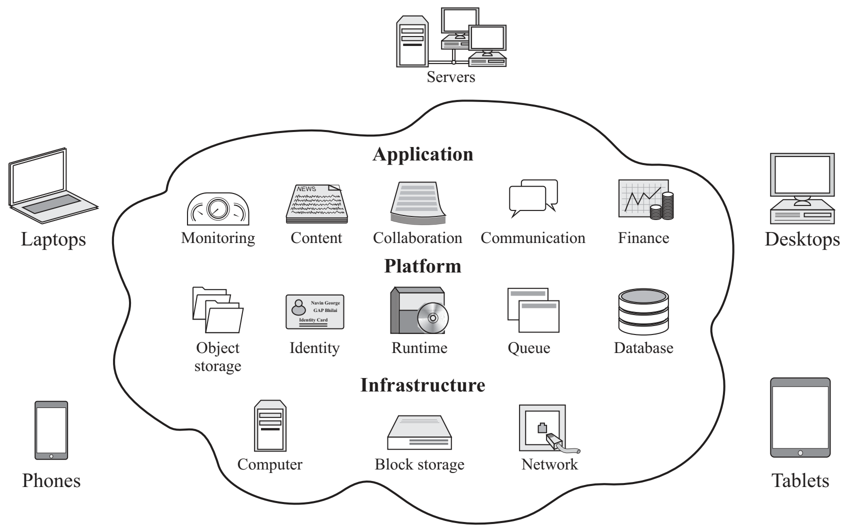


Fig. : Cloud computing

Cloud computing has following five essential characteristics :

1. **On-demand self-services :** Computer services such as e-mail, applications, network or server service can be provided without requiring human interaction with each service provider. Cloud service providers providing on demand self-services include amazon web services (AWS), microsoft, google, IBM and salesforce.com. Newyork times and NASDAQ are examples of companies using AWS.

2. **Broad network access :** Cloud capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms such as mobile phones, laptops and PDAs.

3. **Resource pooling :** The provider's computing resources are pooled together to serve multiple consumers using multiple-tenant model, with different physical and virtual resources dynamically assigned and re-assigned according to consumers demand. The resources include among others storage, processing, memory, network bandwidth, virtual machines and email services. The pooling together of the resource builds economies of scale.

4. **Rapid elasticity :** Cloud services can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

5. **Measured service :** Cloud computing resource usage can be measured, controlled, and reported providing transparency for both the provider and consumer of the utilized service. Cloud computing services use a metering capability which enables to control and optimize resource use. This implies that just like air time, electricity or municipality water IT services are charged per usage metrics **pay per use**. The more you utilize the higher the bill. Just as utility companies sell power to subscribers, and telephone companies sell voice and data services, IT services such as network security management, data center hosting or even departmental billing can now be easily delivered as a contractual service.

Q.7 (b) Discuss the following terms :
(i) Denial of services (ii) Hacking (iii) DBA

Ans. (i) **Denial of services :** DoS is a network based attack whose objective is not to steal the system resources or access confidential data but it aims to prevent the legitimate users from accessing information or services by interrupting the normal use of the system services. DoS attacks fall under two categories :

1. One which eat up almost all system resources, preventing legitimate users from doing any useful work.
2. Another which target the network and disrupt its operation.

The most common type of DoS attack occurs when attackers mischievously flood a network server or a web server with multiple false requests for services in order to crash the network. In this situation, the server is not able to serve the genuine requests. This is a 'denial of service' because the legitimate users cannot use the network facility.

DoS attack does not damage information or access restricted areas but it can shut down a website, thereby making it inaccessible for genuine users. Several times, it becomes difficult for a website to determine that it has been attacked.

Example : A slowdown may be considered as due to network traffic.
It is usually impossible to prevent DoS attacks.

Protecting against DoS attack is as follows :

(i) Make a list of all resource consumed by every user.
(ii) Detect when the resources consumed by a given user exceed those allowed by some system policy.
(iii) After detecting attack, reclaim the consumed resources using as few additional resources as possible or removal of an offending user.

Classification of DoS attacks :

(i) **Logic attacks :** This attack takes place in network software such as TCP/IP protocol stack or web server.
(ii) **Protocol attacks :** Protocol is a set of rules. This attack takes place to specific feature or implementation bug.
(iii) **Bandwidth attacks :** Attacker open many web pages and keep on refreshing for consuming more bandwidth. After some time web site becomes out of service.

Types of DoS attacks :

1. **Ping of death :** Ping of death attack sends large oversized ICMP packets. Maximum legal size of IP packets is 65535 bytes. Because of limitations in the physical layer, packets may have to be fragmented and then reassembled at the destination. So this packet is fragmented for transport. The receiver then starts to reassemble the fragments as the ping fragments arrive. The total packet length becomes too large. It may possible that system may crash.
2. **Smurf :** It is a variation of ping attack. Attacker selects a network of unwitting victims. The attacker spoofs the source address in the ping packet so that it appears to come from the victim. Then the attacker sends this request to the network in broadcast mode by setting the last byte of the address to all 1s.

3. **Teardrop attack :** This attack misuse a feature designed to improve network communication. Attacker sends a series of datagram that cannot fit together properly. One datagram might say it is position 0 for length 60 bytes, another position 30 for 90 bytes so on. These fragment pieces overlap so they cannot be reassembled properly.
4. **Malicious misrouting of packets :** A attacker may attacks a router and change its routing table, resulting in misrouting of data packets, causing a denial of service.
5. Attacker send large number of UDP packets to non-listing ports on the victim. This cause victim to respond with an ICMP host unreachable message for each packet that it receives.

(ii) **Hacking :** Hacking is an art of exploring various security breaches. Each hackers has motives, methods and skills. Computer hacker is a typically knowledgeable person. He/she knows several different languages, familiar with UNIX and NT, Networking protocols. A hackers will look for internal and external system holes or break into the system. Cracker and hacker are two different terms. Cracker is making an attempt to break into the system by guessing or cracking user's passwords. Crackers can easily be identified because their actions are malicious. An ethical hacker possesses the skills, mindset, and tools of a hacker but is also trustworthy. Ethical hackers perform the hacks as security tests for their systems. Ethical hacking is also known as **penetration testing** or **white hat hacking**. It involves the same tools, and techniques that hackers use, but with one major difference. Ethical hacking is legal. Attack vector is a path for hacker. By means which a hacker can gain access to a computer system or computer server in order to deliver a payload or malicious outcome. Attack vectors enable hackers to use system vulnerabilities, including the human. Viruses, attachments of electronic mail, web pages, pop-up windows, instant messages and chat rooms are example of attack vector. All of these methods require programming. To prevent vector, you can use firewall and anti-virus software.

(iii) **Database administrator (DBA) :** The database administrator is a person having central control over data and programs accessing that data. He coordinates all the activities of the database system. The database administrator has a good understanding of the enterprise's information resources and needs.

Functions of a DBA :

1. **Schema definition :** The creation of the original database schema. This involves writing a set of definitions in a DDL (data storage and definition language), compiled by the DDL compiler into a set of tables stored in the data dictionary.
2. **Storage structure and access method definition :** Writing a set of definitions translated by the data storage and definition language compiler.
3. **Schema and physical organization modification :** Writing a set of definitions used by the DDL compiler to generate modifications to appropriate internal system tables (Example : Data dictionary). This is done rarely, but sometimes the database schema or physical organization must be modified.
4. **Granting user authority to access the database :** Granting different types of authorization for data access to various users.
5. **Specifying integrity constraints :** Generating integrity constraints. These are consulted by the database manager module whenever updates occur.
6. **Routine maintenance :** It includes the following :
   (i) Acting as liaison with users.
   (ii) Monitoring performance and responding to changes in requirements.
   (iii) Periodically backing up the database.

❑❑❑