PROGRAMMING FOR PROBLEM SOLVING

1 FUNDAMENTALS OF COMPUTER

PREVIOUS YEARS QUESTIONS

PART A

Q.1 ***What is an algorithm? Write an algorithm to print even numbers from 2 to 100.*** *[R.T.U. 2019]*

Ans. Algorithm : Algorithms are one of the most basic tools that are used to develop the problem-solving logic. An algorithm is defined as a finite sequence of explicit instructions that, when provided with a set of input values, produces an output and then terminates. In algorithm, after a finite number of steps, solution of the problem is achieved. Algorithms can have steps that repeat (iterate) or require decisions (logic and comparison) until the task is completed.

Different algorithms may accomplish the same task, with a different set of instructions, in more or less the same time, space, and efforts. For example, two different recipes for preparing tea, one "add the sugar" while "boiling the water" and the other "after boiling the water" produce the same result. However, performing an algorithm correctly does not guarantee a solution, if the algorithm is flawed or not appropriate to the context. For example, preparing the tea algorithm will fail if there were no tea leaves present, even if all the motions of preparing the tea were performed as if the tea leaves were there. We use algorithms in our daily life. For example, to determine the largest number out of three numbers A, B, and C, the following algorithm may be used.

Step 1: Start
Step 2: Read three numbers say A, B, C
Step 3: Find the larger number between A and B and store it in MAX_AB.
Step 4: Find he larger number between MAX_AB. and C and store it in MAX.
Step 5: Display MAX
Step 6: Stop

The algorithm to print even number from 2 to 100 is as follows.

Step 1 : Start
Step 2 : Initialize NUM ← 2
Step 3 : Execute Steps 4 and 5 Till NUM Equal 100
Step 4 : Calculate NUM ← NUM + 2
Step 5 : Print NUM
Step 6 : Stop

Q.2 ***Explain primary memory and secondary storage.*** *[R.T.U. 2019]*

Ans. Primary memory v/s secondary memory

A computer contains a hierarchy of memory devices for storing data. They vary in their capacity, speed and cost. Primary memory (also referred to as the main memory) is the memory that is directly accessed by the CPU to store and retrieve information. Secondary memory (also referred to as the external or auxiliary memory) is a storage device that is not accessible directly by the CPU and used as a permanent storage device that retains data even after the power is turned off.

Primary memory is the memory that is directly accessed by the CPU to store and retrieve information. Most of the time, primary memory is also referred to as the RAM (Random Access Memory). It is a volatile memory, which loses its data when the power is turned off. Primary memory is directly accessible by the CPU through the address and memory bus and it is constantly accessed by the CPU to get data and instructions. Furthermore, computers contain a ROM (Read Only

Memory), which holds instructions that are executed often such as the startup program (BIOS). This is a non volatile memory that retains its data when the power is turned off. Since the main memory is accessed often, it needs to be faster. But they are smaller in size and also costly.

Secondary memory is a storage device that is not accessible directly by the CPU and used as a permanent storage device that retains data even after power is turned off. CPU accesses these devices through an input/ output channel and data is first transferred in to the primary memory from the secondary memory before accessing. Usually, hard disk drives and optical storage devices (CDs, DVDs) are used as secondary storage devices in modern computers. In a secondary storage device, data are organized in to files and directories according to a file system. This also allows to associate additional information with data such as the access permissions, owner, last access time, etc. Furthermore, when the primary memory is filled up, secondary memory is used as a temporary storage for keeping least used data in the primary memory. Secondary memory devices are less costly and larger in size. But they have a large access time.

However, primary memory is the memory that is directly accessed by the CPU to store and retrieve information, whereas the secondary memory is not accessible directly by the CPU. Primary memory is accessed using address and data buses by the CPU, while secondary memory is accessed using input/ output channels. Primary memory does not retain data when the power is turned off (volatile) while secondary memory retains data when the power is turned off (non-volatile). Furthermore, primary memory is very fast compared to the secondary memory and has a lower access time. But, primary memory devices are more costly compared to secondary memory devices. Due to this reason, usually a computer comprises of a smaller primary memory and a much larger secondary memory.

Q.3 ***Write the difference between random access and sequential access method.*** ***[R.T.U. 2019]***

Ans. Random Access Method : In computer science, **random access** (sometimes called **direct access**) is the ability to access an arbitrary element of a sequence in equal time. The opposite is sequential access, where a remote element takes longer time to access. A typical illustration of this distinction is to compare an ancient scroll (sequential; all material prior to the data needed must be unrolled) and the book (random: can be immediately flipped open to any random page). A more modern example is a cassette tape (sequential—you have to fast-forward through earlier songs to get to later ones) and a compact disc (random access—you can jump right to the track you want). The term Random Access Memory (RAM), however, is used for semiconductor chip memory circuits used in computers. (The term was also used to describe ferrite-core memory in early computers).

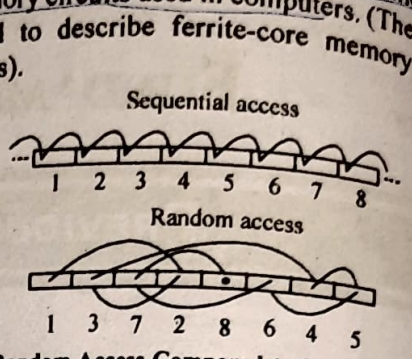


Fig. : Random Access Compared to Sequential Access

In data structures, random access implies the ability to access the n^{th} entry in a list of numbers in constant time. Very few data structures can guarantee this, other than arrays (and related structures like dynamic arrays). Random access is critical to many algorithms such as quick sort and binary search. Other data structures, such as linked lists, sacrifice random access to make for efficient inserts, deletes, or reordering of data.

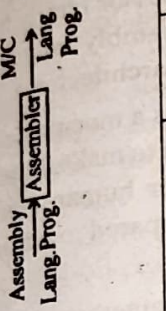
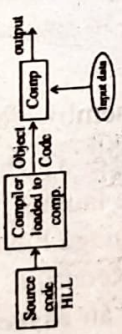
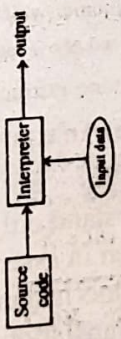
Sequential Access Method

In computer science, **sequential access** means that a group of elements (e.g. data in a memory array or a disk file or on a tape) is accessed in a predetermined, ordered sequence. Sequential access is sometimes the only way of accessing the data, for example if it is on a tape. It may also be the access method of choice, for example if we simply want to process a sequence of data elements in order.

In data structures, a data structure is said to have sequential access if one can only visit the values it contains in one particular order. The canonical example is the linked list. Indexing into a list which has sequential access requires $O(k)$ time, where k is the index. As a result many algorithms such as quicksort and binary search degenerate into bad algorithms that are even less efficient than their alternatives; these algorithms are impractical without random access. On the other hand, some algorithms, typically those which don't perform indexing, require only sequential access, such as mergesort, and so face no penalty.

Q.4 ***Write the difference between compiler and interpreter.*** ***[R.T.U. 2019]***

Ans.

Assembler	Compiler	Interpreter
1. Assembler is a translator that translates the assembly language program into the m/c language. 	1. Complex and difficult to write	1. Simple and easy to write
	2. Require large memory space to store itself	2. Doesn't require much memory space to store itself.
	3. Takes more time in translation.	3. Take less time in translation
	4. Translates the entire program into m/c language at once.	4. Translates the program line by line into m/c language
2. It also assembles the m/c code in memory of computer and makes the program ready for execution.	5. Checks the syntax errors in program when whole program become error free then it is translated into m/c code and executed	5. Check for the syntax errors line by line and executes each Line immediately if statement is error free.
	6. Takes less time to execute program	6. Takes more time to execute programs
	7. Slow for debugging 	7. Fast for debugging 

Q.5 Write the difference between High-Level language and Low-Level language. [R.T.U. 2018]

Ans. High level v/s Assembly level programming :

	Machine language	Assembly language	High level langnage
1.	It is the language *i.e.* understands by the computer without using a translation program.	In assembly language we use mnemonic for numeric opcodes of m/c language	High level language is basically problem oriented. While writing program in the HLL a programmer use commands generally written using English words and mathematical symbols. HLL programs are translated into machine language by means of a system program called compilers.
2.	This is also called as m/c code of the computer and is normally written as string of binary 0's and 1's.	Computer can translate each Line of assembly language Program into m/c language. program by means of translating program.	
3.	**Advantages** Program in m/c language can be executed very fast.	**Advantages** Easy to understand and use easier to modify & relocatable.	**Advantages** Machine independent fewer errors and easy debugging.
4.	**Disadvantages** 1. Machine dependent. 2. Difficult to program and modify. 3. Time consuming and error prone.	**Disadvantages** 1. Machine dependent. 2. Knowledge of h/w required.	**Disadvantages** Speed (HLL program takes more time and memory to run).

Q.6 Write short note on Pseudo Code. [R.T.U. 2015, 13]

Ans. Pseudocode : Pseudocode is made up on two words 'Pseudo' and 'Code'. Pseudo means 'false' or

'unitation' and code means 'instruction' written in a programming language. As the name suggests, pseudocode is not a real programming code, but it models programming code. It is generic way of describing an algorithm without using any specific programming language constructs. Pseudocode uses plain english statement rather than symbols to represent the process of a computer program.

Advantages of Pseudocode

(i) Since it is language independent, it allows the developer to express the design in plain natural language.

(ii) Unlike flowcharts, pseudocode is compact and does not tend to run over many phases. Its simple structure and readability makes it easier to modify as well.

Limitations

(i) The main limitation of pseudocode is that it does not provide visual representation of the program's logic.

(ii) There are no accepted standards for writing pseudocodes. Different programmers use their own style of writing pseudocode.

Q.7 ***Write short note on Flow chart and algorithms.*** *[R.T.U. 2015]*

Ans. Flowchart : A flowchart is a pictorial representation of an algorithm in which the steps are drawn in the form of different shapes of boxes and the logical flow is indicated by interconnecting arrows. The boxes represent operations and the arrows represent the sequence in which the operations are implemented. The primary purpose of the flowchart is to help the programmer in understanding the logic of the program. Therefore, it is always not necessary to include all the required steps in detail. Flowcharts outline the general procedure. Since they provide an alternative, visual way of representing the information flow in a program, program developers often find them very valuable.

Flowcharts can be compared with the blueprint of a building. Just as an architect draws a blueprint before starting the construction of a building, a programmer draws a flowchart before writing a computer program. As in the case of the drawing of a blueprint, the flowchart is drawn according to defined rules and using standard flowchart symbols prescribed by American National Standard Institute (ANSI).

Algorithms : *Refer to Q.1.*

Q.8 ***Write short note on assembly language.*** *[R.T.U. 2013]*

Ans. Assembly Language : The complexities of the machine language led to the development of Assembly Language which is called second generation of programming languages. Unlike other programming languages, assembly language is not a single language, but a group of languages. Each processor family has its own assembly language. Thus, assembly language is different for computers of different architectures.

The assembly language assigns a mnemonic code to each machine language instruction to make it easier to remember or write. It allows better human readable method of writings programs as compared to writing in binary bit patterns.

Assembly language provides mnemonics i.e., is easy to remember short names, for the various operations supported by the machine language. For example, ADD is used to perform addition operation, MULT is used for multiplication etc. The assembly language also allows us to refer to memory locations by symbolic names. Thus an assembly language program consists of symbols or names instead of numeric codes.

Apart from the use of mnemonics and symbolic names, the assembly language instructions have one to one correspondence with the machine language instructions. A sample machine language instruction and its equivalent assembly language instruction.

1	0	1	0	0	0	0	1	1	0	0	1

Operation code Address of a memory location

(a) A sample machine language instructions

ADD	NUM
Mnemonic, which has replaced the opcode in machine instruction.	A symbolic name which has replaced the address of a location in the machine instruction.

(b) An equivalent Assembly language instruction

As you are aware of that a computer can understand only machine language, hence the programs written in assembly language have to be converted into machine language for execution. A program that does this translation is called an **assembler**. The assembler recognizes the assembly language mnemonics and symbolic names of memory locations and converts them into appropriate binary code.

Q.9 What do you mean by wordlength of a processor. *[R.T.U. 2012]*

Ans. Wordlength is the number of bits that a CPU can process at once. A processor with a 32-bit wordlength has the capacity to be twice as fast as a processor with a 16 bit wordlength. Today's personal computers typically use wordlengths of 32 or 64 bits. Intel Pentium 4 processors have a 32 bit wordlength; AMD Athlon 64 processors have a word length of 64 bits. The benefits of 64 bit wordlength are only experienced if the operating system and software are designed to take advantage of the processor's 64-bit capabilities. While Intel processors and both Windows and Mac OS X have evolved to 64 bit architectures, software developers have been slower to migrate their applications from 32-bit to 64-bit. That is likely to change as 64-bit processors become the norm.

PART B

Q.10 Write a Pseudo code to multiply 2[3 × 3] Matrices and Transpose the output of multiplication. *[R.T.U. 2019]*

Ans.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,m,n,p,q,k,sum=0; /* Declaring sum=0 since we use it as a temporary variable while multiplication */
int first[3][3], second[3][3], multiply[3][3], transpose[3][3];
/* Accepting 1st matrix */
printf("Enter the number of rows and columns of first matrix\n");
scanf("%d%d",&m,&n);
printf("Enter the elements of first matrix\n");/* Order (m,n) */
 for(i=0;i<m;i++)
   {
  for(j=0;j<n;j++)
     {
     scanf("%d",&first[i][j]);
     }
   }
/* Accepting 2nd Matrix */
printf("Enter the number of rows and columns of second matrix\n");
scanf("%d%d",&p,&q);
if(n!=p) /* To multiply matrices, Number of columns of 1st matrix should be equal to number of rows in 2nd matrix */
    printf("matrix with entered order can't be multiplied with each other.\n");
else /* Accepting 2nd Matrix */
{
printf("Enter the element of second matrix\n");
/* Order (p,q) */
for(i=0;i<p;i++)
{
  for(j=0;j<q;j++)
   {
     scanf("%d",&second[i][j]);
   }
}
/*Matrix Multiplication*/
     for(i=0;i<m;i++)
       {
       for(j=0;j<q;j++)
         {
        for(k=0;k<p;k++)
          {
          sum=sum+first[i][k]*second[k][j];
/* calculating and storing final value of element in sum */
          }
multiply[i][j]=sum; /* Assigning value stored in sum to Matrix: multiply[i][j] */
     sum=0; /* Again we will make sum=0 since we want to multiply next element */
          }
       }
/* Printing multiplied matrix */
printf("Product of entered matrices\n");
for(i=0;i<m;i++)
  {
  for(j=0;j<q;j++)
    {
    printf("%d\t",multiply[i][j]);
```

```
        }
    printf("\n");
      }
    }
    /* transpose of matrix */
    for(i=0;i<m;i++)
    {
      for(j=0;j<n;j++)
      {
        transpose[i][j]=multiply[j][i];
    /*for example i=1 and j=2 then in matrix, element
(1,2) will be swapped with (2,1) thus we get transpose of
matrix*/
      }
    }
    /* Printing transposed matrix */
    printf("Transpose of the matrix \n");
    for(i=0;i<n;i++)
    {
      for(j=0;j<m;j++)
      {
        printf("%d\t",transpose[i][j]);
      }
      printf("\n"); /* \n to bring cursor to next line and
print elements in matrix format */
    }
    getch();
    }
```

Q.11 ***Draw the block diagram of a Computer. Explain its components. Differentiate between primary and secondary storage.*** ***[R.T.U. Dec. 2019]***

OR

Draw a block diagram of basic architecture of a computer system. ***[R.T.U. May 2019]***

Ans. The size, shape, performance, reliability, and cost of computers have been changing over the years. The basic logical structure (based on the stored program concept), as proposed by Von Neumann, has not changed. No matter what shape and size of computer we are talking about.

All computer systems perform the following five basic operations, for converting raw input data into information which is useful to their users :

1. **Inputting :** The process of entering data and instructions into the computer system.
2. **Storing :** Saving data and instructions to make them readily available for initial or additional processing, as and when required.

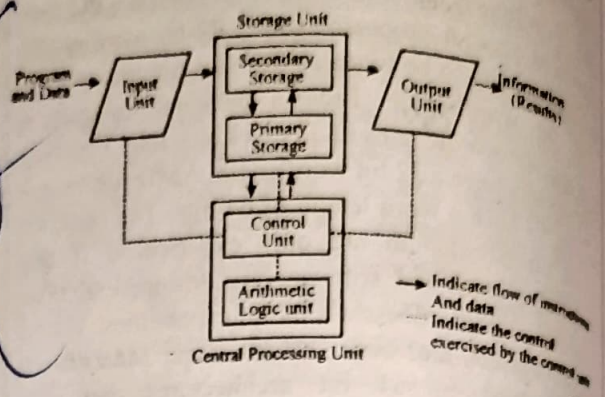


Fig. : Architecture of a computer system

3. **Processing :** Performing arithmetic operations (add, subtract, multiply, divide, etc.), or logical operations (comparisons like equal to, less than, greater than, etc.) on data, to convert them into useful information.
4. **Outputting :** The process of producing useful information or result for the user, such as a printed report or visual display.
5. **Controlling :** Directing the manner and sequence in which all of the above operations are performed.

The internal architecture of computers differs from one system model to another. However, the basic organization remains the same for all computer systems. A block diagram of the basic computer organization is shown in figure. In the given figure, the solid lines indicate the flow of instruction and data, and the dotted lines represent the control exercised by the control unit. It displays the five major building blocks (functional units) of a digital computer system. These five units correspond to the five basic operations, performed by all computer systems.

Input Unit : Data and instructions must enter the computer system, before any computation is performed on the supplied data. This task is preformed by the input unit, which links the external environment with the computer system. Data and instructions enter input units in forms, which depend upon the particular device used. For example, data are entered from a keyboard in a manner similar to typing, and this differs from the way in which data are entered through a scanner, which is another type of input device. However, regardless of the form in which they receive their inputs, all input devices

must transform the input data into the binary codes, which the primary memory of a computer is designed to accept. This transformation is accomplished by units called input interfaces. Input interfaces are designed to match the unique physical or electrical characteristics of input device to the requirements of the computer system.

In short, an input unit performs the following functions :

1. It accepts (or reads) the instructions and data from the outside world.
2. It converts these instructions in computer acceptable form.
3. It supplies the converted instructions and data to the computer system for further processing.

Output Unit : The job of an output unit is just the reverse of that of an input unit. It supplies the information obtained from data processing, to the outside world. Hence, it links the computer with the external environment. As computers work with binary code, the results produced are also in the binary form. Hence before supplying the result to the outside world, they must be converted to human acceptable (readable) form. This task is accomplished by units called output interfaces. Output interfaces are designed to match the unique physical or electrical characteristics of output devices (terminals, printers, etc.), to the requirements of the external environment.

In short, an output unit performs the following functions :

1. It accepts the results produced by the computer, which are in coded form, and hence cannot be easily understood by us.
2. It converts these coded results to human acceptable (readable) form.
3. It supplies the converted result to the outside world.

Storage Unit : The data and instructions, which are entered into the computer system through input units, have to be stored inside the computer, before the actual processing starts. Similarly, the results produced by the computer after processing, must also be kept somewhere inside the computer system, before being passed on to the output units. Moreover, the intermediate results produced by the computer, must also be preserved for ongoing processing. The storage unit of a computer system is designed to cater to all these needs. It provides space for storing data and instructions, space for intermediate results, and space for the final results.

In short, the specific functions of the storage unit are to hold (store):

1. The data and instructions required for processing (received from input devices).
2. Intermediate results of processing.
3. Final results of processing, before these results are released to an output device.

The storage unit of all computers is comprised of the following two types of storage :

1. Primary Storage : The primary storage, also known as main memory, is used to hold pieces of program instructions and data, intermediate results of processing, and recently produced results of processing, of the job(s), which the computer system is currently working on. These pieces of information are represented electronically in the main memory chip's circuitry, and while it remains in the main memory, the central processing unit can access it directly at a very fast speed. However, the primary storage can hold information only while the computer system is on. As soon as the computer system is switched off or reset, the information held in the primary storage disappears. Moreover, the primary storage normally has limited storage capacity, because it is very expensive. The primary storage of modern computer system is made up of semiconductor devices.

2. Secondary Storage : The secondary storage, also known as auxiliary storage, is used to take care of the limitations of the primary storage. That is, it is used to supplement the limited storage capacity and the volatile characteristic of primary storage. This is because secondary storage is much cheaper than primary storage, and it can retain information even when the computer system is switched off or reset. The secondary storage is normally used to hold the program instructions, data, and information of those jobs, on which the computer system is not working on currently, but needs to hold them for processing later. The most commonly used secondary storage medium is the magnetic disk.

Central Processing Unit (CPU) : The Control Unit and the Arithmetic Logic Unit of a computer system are jointly known as the Central Processing Unit (CPU). The CPU is the brain of a computer system. In a human body, all major decisions are taken by the brain, and the other parts of the body function are directed by the brain. In a computer system, all major calculations and comparisons are made inside the CPU, and the CPU is responsible for activating and controlling the operations of other units of the computer system.

(i) Arithmetic Logic Unit : The Arithmetic Logic Unit (ALU) of a computer system is the place, where the actual execution of the instructions takes place, during the processing operation. To be more precise, calculations are performed, and all comparisons (decision) are made in the ALU. The data and instructions, stored in the primary storage before processing, are transferred as and when needed to the ALU, where processing takes place. No processing is done in the primary storage unit. Intermediate results generated in the ALU are temporarily transferred back to the primary storage, until needed later. Hence, data may move from primary storage to ALU, back again to storage, many times, before the processing is over.

The type and number of arithmetic and logic operations, which a computer can perform, is determined by the engineering design of the ALU. However, almost all ALUs are designed to perform the four basic arithmetic operations (add, subtract, multiply, and divide), and logic operations or comparisons (such as less than, equal to, and greater than).

(ii) Control Unit : How does the input device known that it is time for it to feed data into the storage unit? How does the ALU know, what should be done with the data once they are received ? Moreover, how is it that only the final results are sent to the output device, and not the intermediate results? All this is possible due to the control unit of the computer system. Although, it does not perform any actual processing on the data, the control unit acts as a central nervous system, for the other components of the computer system. It manages and coordinates the entire computer system. It obtains instructions from the program stored in main memory, interprets the instructions, and issues signals, which cause other units of the system to execute them.

Q.12 ***Write short notes on :***

(a) System software

(b) Compiler

(c) Freeware software ***[R.T.U. 2018]***

Ans.(a) System Software : System software is a type of computer program that is designed to run a computer's hardware and application programs. If we think of the computer system as a layered model, the system software is the interface between the hardware and user applications.

The operating system (OS) is the best-known example of system software. The OS manages all the other programs in a computer. Other examples of system software and what each does:

- The BIOS (basic input/output system) gets the computer system started after you turn it on and manages the data flow between the operating system and attached devices such as the hard disk, video adapter, keyboard, mouse, and printer.
- The boot program loads the operating system into the computer's main memory or Random Access Memory (RAM).
- An assembler takes basic computer instructions and converts them into a pattern of bits that the computer's processor can use to perform its basic operations.
- A device driver controls a particular type of device that is attached to your computer, such as a keyboard or a mouse. The driver program converts the more general input/output instructions of the operating system to messages that the device type can understand.

According to some definitions, system software also includes system utilities, such as the disk defragmenter and system restore, and development tools such as compilers and debuggers.

Ans.(b) Compiler : A compiler is a kind of translator that translates a program written in a high level language into machine code. The compiler replaces single high level statement into a series of machine language instructions. During the translation process, the compiler reads the source program and checks the syntax (grammatical) errors. If there is any error, the compiler generates an error message which is displayed on the screen. In case of errors, the compiler doesn't create the object code until all the errors are rectified.

Once the source code has been compiled successfully, the compiler saves the resulting machine code in a file separately. This make recompiling necessary only if the source code has been modified.

The working of a compiler can be depicted as :

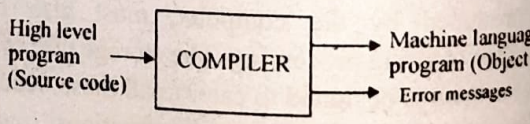


Fig. : Working of a Compiler

Ans.(c) Freeware Software : Freeware is computer software that is made available free of charge, but which is copyrighted by its developer, who retains the rights to control its distribution, modify it and sell it in the future. It is typically distributed without its source code, thus preventing modification by its users.

This can be freely used with only one restriction: any redistributed version of the software must be distributed with the original terms of free use, modification, and distribution (known as copyleft). Freeware is usually distributed with a license that permits its redistribution to some extent, for example allowing users to give copies to friends. Some licenses permit the software to be freely copied but not sold.

Q.13 ***Give classification for the types of language working in computer programming environment.*** ***[R.T.U. 2015]***

Ans. Computers understand only one language and that is binary language or the language of '0's and 'l's. Binary language is also known as machine language. In the initial years of computer programming, all the instructions were given in binary form only. Although these programs were easily understood by the computer, it proved too difficult for a normal human being to remember all the instructions in the form of '0's and 'l's. Therefore, the computer remained a mystery to a common person until other languages, such as assembly and high-level languages, were developed which were easier to learn and understand. These languages use commands that have some degree of similarity with English (such as "if else," "exit"). Programming languages can be divided into three major categories:

1. Machine Language : It is the native language of computers. It uses only '0's and 'l's to represent data and the instructions.

2. Assembly Language: It corresponds to symbolic instructions and executable machine codes and was created to use letters instead of '0's and 'l's to run a machine.

3. High-level Language: These languages are written using a set of words and symbols following some rules similar to a natural language, such as English. The programs written in high-level languages are known as source programs and these programs are converted into machine-readable form by using compilers or interpreters.

Note : Together, machine and assembly language are also known as low-level languages.

Q.14 ***Write differences between the following :***
(i) Interpreter and compiler
(ii) High level, assembly and machine language. ***[R.T.U. 2014]***

OR

Machine, Assembly and High Level Language ***[R.T.U. 2011]***

Ans.(i) *Refer to Q.4.*
(ii) *Refer to Q.5.*

Q.15 ***Differentiate between random access method and sequential access method for memory devices.*** ***[R.T.U. 2014, 2012]***

Ans. Random Access Method : *Refer to Q.3.*

Q.16 ***(a) What is pseudocode ? Write the advantage and limitation of pseudocode.*** ***[R.T.U. 2010]***
(b) Write short note on volatile memory. ***[R.T.U. 2013]***

OR

Write short note on direct access memory. ***[R.T.U. 2013]***

Ans.(a) Pseudocode : *Refer to Q.6.*

Ans.(b) Volatile Memory : It requires constant power to maintain the stored information. The fastest memory technologies of today are volatile ones. Since primary storage is required to be very fast, it predominantly uses volatile memory.

Random Access Memory (RAM) : RAM is volatile in nature, this memory consists of some integrated circuit chips either on motherboard, or on a small circuit board attached to the motherboard. A computer's motherboard is designed in a manner that its memory capacity can be easily enhanced by adding more memory chips. If you require to insert more memory, then you can easily plug-in extra memory in the empty memory slots on the motherboard. A computer containing only primary memory would not have a source to read instructions, in order to start the computer. Hence, non-volatile primary memory containing a small startup program (BIOS) is used to bootstrap the computer, that is, to read a larger program from non-volatile secondary memory to RAM and start to execute it.

The functioning of an interpreter can be shown as:

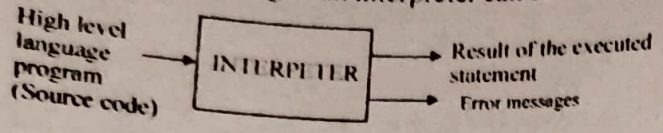


Fig. 2 : Working of an Interpreter

Q.20 ***Differentiate between ROM and PROM.***

[R.T.U. 2011, Raj. Univ. 2002]

Ans. Difference between ROM and PROM

ROM (Read Only Memory) : A special type of RAM, called Read Only Memory (ROM), is a non–volatile memory chip, in which data is stored permanently and cannot be altered by the programmer. In fact, storing data permanently into this kind of memory is called "burning in the data", because data in such memory is stored by using fuse–links. Once a fuse–link is burnt, it is permanent. The data stored in a ROM chip can only be read and used, they cannot be changed. This is the reason why it is called Read Only Memory (ROM). Since ROM chips are non–volatile, the data stored inside a ROM are not lost when the power supply is switched off, unlike the case of a volatile RAM chip. ROMs are also known as field stores, permanent stores, or dead stores.

ROMs are mainly used to store program and data, which do not change and are frequently used. For example, the most basic computer operations are carried out by wired electronic circuits. However, several higher–level operations, which are frequently used, require very complicated circuits for their implementations.

Hence, instead of building electronic circuits for these operations, special programs are written to perform them. These programs are called micro programs, because they deal with low–level machine functions and are essentially substitutes for additional hardware. ROMs are used by computer manufacturers for storing these micro programs, so that they cannot be modified by the users.

PROM (Programmable Read Only Memory) : Similar to Read Only Memory with the exception that these chips can be reprogrammed by using special external equipments. There are two type of Read Only Memory (ROM)-manufacturer-programmed and user-programmed.

A **manufacturer-programmed** ROM is one in which data is burnt in by the manufacturer of the electronic equipment in which it is used. For example, a personal computer manufacturer may store the system boot program permanently in the ROM chip used on the motherboard of all the PCs manufactured by it. Similarly, a printer manufacturer may store the printer controller software in the ROM chip used on the circuit board of all the printers manufactured by it. **Manufacturer-programmed** ROMs are mainly used in those cases where the demand for such programmed ROMs is large. Note that manufacturer programmed ROM chips are supplied by the manufacturers of electronic equipment, and it is not possible for a user to modify the programs or data stored inside the ROM chip. On the other hand, a user-programmed ROM is one in which the user can load and store "read only" programs and data. That is, it is possible for a user to "customize" a system by converting his/her own programs to micro programs, and storing them in a user-programmed ROM chip. Such a ROM is commonly known as Programmable Read Only Memory (PROM), because a user can program it. Once the user programs are stored in a PROM chip, they can usually be executed in a fraction of the time previously required. PROMs are programmed to record information using a special device, known as PROM-programmer. However, once the chip has been programmed, the recorded information cannot be changed, *i.e.*, the PROM become a ROM, and it is only possible to read the stored information. PROM is also non-volatile storage, *i.e.*, the stored information remains intact, even if power is switched off.

Q.21 ***Write short note on different types of RAMs.***

[R.T.U. 2010, Raj. Univ. 2005]

Ans. Random Access Memory (RAM) : *Refer to Q.16(b).*

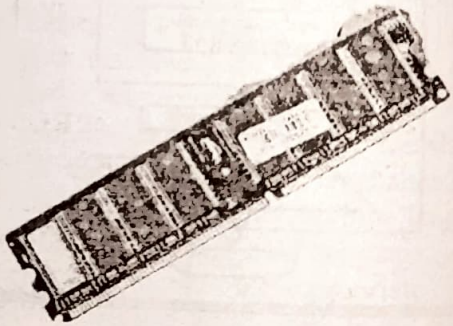


Fig. : RAM Chip

Types of RAM :

Primary Memory : Random Access Memory (RAM) and Read Only Memory (ROM) fall in the category of

the primary memory. Every computer comes with a small amount of ROM, which contains the boot firmware (called BIOS). This holds just enough information so that the computer can check its hardware and load its operating system into RAM.

RAM is the place where the computer temporarily stores its operating system, application programs, the current data, so that the computers processor can reach them quickly and easily. It is volatile in nature, *i.e.* when the power is switched off, the data in the memory is lost. Unlike RAM, ROM is non-volatile. Even when the computer is switched off, the contents of ROM remain available.

Types of RAM : RAM can be further divided into two categories :

(i) Static Random Access Memory (SRAM) is a type of semiconductor memory that does not need to be periodically refreshed like DRAM. SRAM is considerably faster than DRAM. It is effective because most of the programs access the same data repeatedly and keeping all this information in the fast SRAM that allows the computer to avoid accessing the slow DRAM. SRAM exhibits remembrance, but is still volatile in the conventional sense that data is eventually lost when the memory is not powered.

(ii)Dynamic Random Access Memory (DRAM): It is a type of RAM that stores each bit of data in a separate capacitor within an integrated circuit. Since real capacitor leak charge, the information eventually fades unless the capacitor charge is refreshed periodically. Because of this refresh requirement, it is a dynamic memory as opposed to SRAM and other static memory. Like SRAM, it is in the class of volatile memory devices, since it loses its data when the power supply is removed. Unlike SRAM however, data may still be recovered for a short time after power-off.

Types of ROM -

Programmable read-only memory (PROM), or onetime programmable ROM (OTP), can be written to or programmed via a special device called a PROM programmer. Typically, this device uses high voltages to permanently destroy or create internal links (fuses or antifuses) within the chip. Consequently, a PROM can only be programmed once.

Erasable programmable read-only memory (EPROM) can be erased by exposure to strong ultraviolet light (typically for 10 minutes or longer), then rewritten with a process that again requires application of higher than usual voltage. Repeated exposure to UV light will eventually wear out an EPROM, but the endurance of most EPROM chips exceeds 1000 cycles of erasing and reprogramming. EPROM chip packages can often be identified by the prominent quartz "window" which allows UV light to enter. After programming, the window is typically covered with a label to prevent accidental erasure.

Electrically erasable programmable read-only memory (EEPROM) is based on a similar semiconductor structure to EPROM, but allows its entire contents (or selected banks) to be electrically erased, then rewritten electrically, so that they need not be removed from the computer (or camera, MP3 player, etc.). Writing or flashing an EEPROM is much slower (milliseconds per bit) than reading from a ROM or writing to a RAM (nanoseconds in both cases).

Electrically alterable read-only memory (EAROM) is a type of EEPROM that can be modified one bit at a time. Writing is a very slow process and again requires higher voltage (usually around 12 V) than is used for read access. EAROMs are intended for applications that require infrequent and only partial rewriting. EAROM may be used as non-volatile storage for critical system setup information; in many applications, EAROM has been supplanted by CMOS RAM supplied by mains power and backed-up with a lithium battery.

Flash memory (or simply flash) is a modern type of EEPROM invented in 1984. Flash memory can be erased and rewritten faster than ordinary EEPROM, and newer designs feature very high endurance (exceeding 1,000,000 cycles). Modern NAND flash makes efficient use of silicon chip area, resulting in individual ICs with a capacity as high as 16 GB as of 2007[update]; this feature, along with its endurance and physical durability, has allowed NAND flash to replace magnetic in some applications (such as USB flash drives). Flash memory is sometimes called flash ROM or flash EEPROM when used as a replacement for older ROM types, but not in applications that take advantage of its ability to be modified quickly and frequently.

Q.22 ***Discuss various types of memory. Explain primary and secondary memory in detail.*** ***[R.T.U. 2010, 2008]***

Ans. Memory refers to the electronic holding place for instructions and data. CPU requires memory to handle the initial input, intermediate and final results. We can classify memory into two basic categories :

(i) Primary memory

(ii) Secondary memory

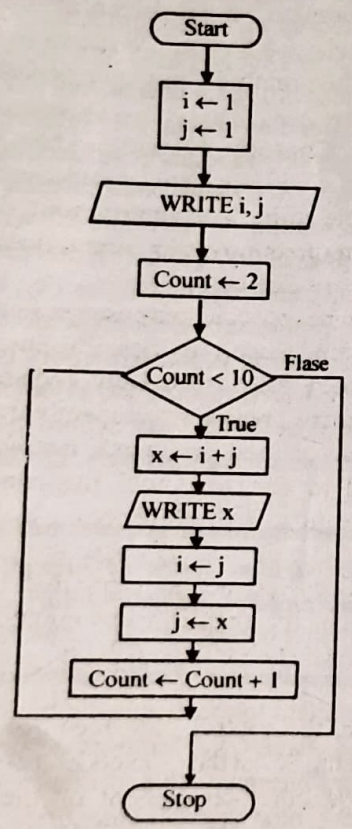
(i) Primary Memory : *Refer to Q.21.*

(ii) Secondary Memory : It is also known as auxiliary memory. Secondary memory provides backup storage for instructions (software programs) and data. Most commonly used secondary storage devices are Hard Disk, Magnetic Disk and Magnetic Tapes. These are the least expensive among all the memories. However, they have much larger storage capacity than primary memory. Instructions and data stored on such devices are permanent in nature. It can only be removed if the user wants it or the device is destroyed the benefits of secondary storage are :

(i) Non volatility
(ii) Reliability
(iii) Convenience
(iv) Less expensive
(v) Reusability
(vi) Portability

Q.23 ***Draw a flow chart for computing first 10 fibonacci number.*** *[R.T.U. 2009]*

Ans. Flow Chart of Fibonacci Number



PART C

Q.24 ***Write short notes on following :***

(a) Types of Primary Memory
(b) High level v/s Assembly level programming
(c) Notations of Flow chart
(d) Primary memory v/s secondary memory

[R.T.U. 2014]

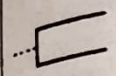
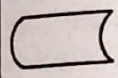
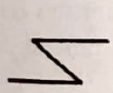
Ans.(a) Primary Memory : *Refer to Q.21.*

Ans.(b) High level v/s Assembly level programming : *Refer to Q.5.*

Ans.(c) Notations of Flowchart

There are rules and standards (rotations) for drawing flowcharts, prescribed by American Nation... Standard Institute (ANSI). Table shows some standard symbols which are frequently required in the flow charts.

Symbol	Symbol Name	Description
	Flow lines	Flow lines are used to connect symbols used in flow-chart and indicate direction of flow.
	Terminal (START / STOP)	This is used to represent start and end of the flow-chart.
	Input / Output	It represent information which the system reads as input or sends as output.
	Processing	Any process is represented by this symbol. For example arithmetic operation, data movement.
	Decision	This symbol is used to check any condition or take decision for which there are two answers. Yes (True) or No (False).

	Connector	It is used to connect or join flow lines.
	Off-page Connector	This symbol indicates the communication of flowchart on the next page.
	Document	It represents a paper document produced during the flowchart process.
	Annotation	It is urged to provide additional information about another flowchart symbol which may be in the form of descriptive comments, remark or explanatory notes.
	Manual Input	It represents input to be given by a developer or programmer.
	Manual operation	This symbol indicates that the process has to be done by a developer or programmer.
	Online storage	It represents online data storage such as hard disks, magnetic drums or other storage devices.
	Offline storage	It represents the offline data storage such as sales on OCR, data on punched card.
	Communication Link	It represents the data received or to be transmitted from an external system.
	Magnetic Disk	It represent data input or output from and to a magnetic disk.

Ans.(d) Primary memory v/s secondary memory : *Refer to Q.2.*

Q.25 ***Draw a block diagram of basic architecture of a computer system.*** ***[R.T.U. 2014]***

OR

Design and explain the architecture of computer. ***[R.T.U. 2011]***

Ans. *Refer to Q.11.*

Q.26 ***What is the function of CPU in a computer system?*** ***[R.T.U. 2010, 2007]***

Ans. The Central Processing Unit (CPU) : The CPU is the brain of a computer system. All major calculations and comparisons performed by a computer are carried out inside its CPU. The CPU is also responsible for activating and controlling the operations of the computer system. Hence, no other single component of a computer determines its overall performance, as much as the CPU. In order to be able to quickly evaluate any computer's capabilities, it is important to know how CPUs are internally structured, how different CPUs differ from each other, and how CPU speed is evaluated. These and other related concepts about CPU are described below :

(i) The Control Unit : The two basic component of a CPU are the control unit and the arithmetic logic unit. The control unit of the CPU selects and interprets program instructions, and then sees that they are executed.

It has some special purpose registers, and a decoder to perform these activities. The special purpose registers, namely the **instruction register** and the **program control register**, respectively hold the current instruction and the next instruction to be executed, and in this way help the control unit in instruction selection. On the other hand, the decoder has the necessary circuitry to decode and interpret the meaning of every instruction supported by the CPU. Each instruction is accomplished by microcode *i.e.* very basic directions, which tell the CPU how to execute the instruction.

Although, the control unit does not perform any actual processing of the data, it acts as a central nervous system for the other component of the computer it manages and coordinates the entire computer system, including the input and output units. It obtains instructions

from the program stored in the main memory, interprets the instructions, and issues signals, which cause other units of the system to execute them.

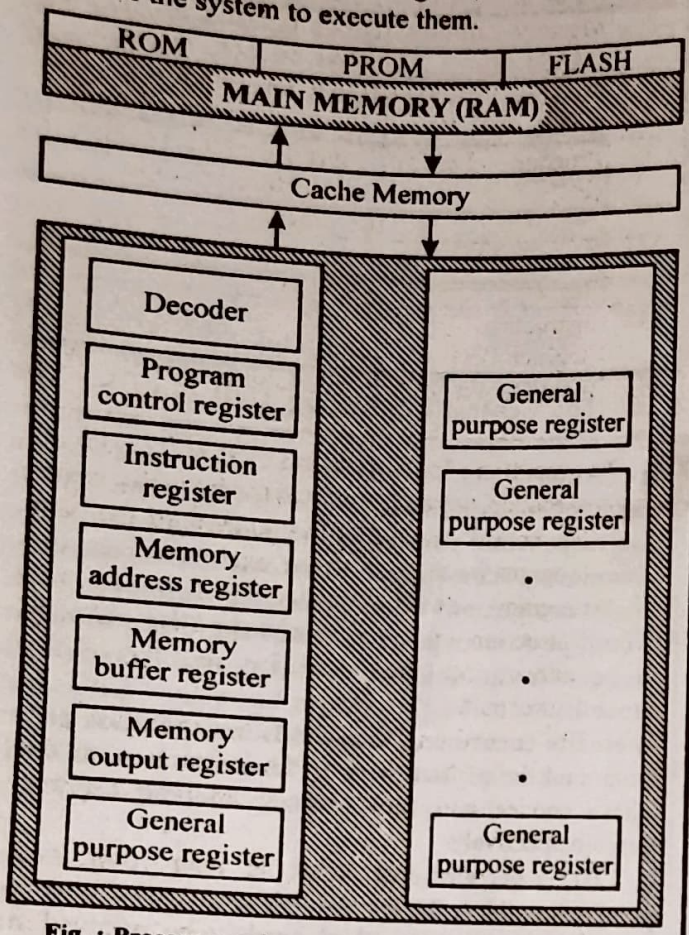


Fig. : Processor and memory architecture of a computer system

(ii) The Arithmetic Logic Unit (ALU) : The ALU of the CPU is the place, where the actual execution of the instructions takes place, during the data processing operation. That is, when the control unit encounters an instruction, which involves an arithmetic operation (such as, add, subtract, multiply, divide), or a logic operation (such as less than, equal to, greater than), it passes control to the ALU. The ALU has some special purpose registers and the necessary circuitry, to carry out all the arithmetic and logic operations, which are included in the instructions supported by the CPU. For example, the control unit might load two numbers into the registers in the ALU. Then, it might tell the ALU to add the two numbers (an arithmetic operation), or to check if the two numbers are equal (a logical operation).

In case of a microcomputer, the entire CPU (both the control unit and the ALU) is contained on a single tiny silicon chip, called a microprocessor.

Instruction Set : Every CPU has built–in ability to execute a set of machine instructions, called its instruction set. Most CPUs have 200 or more instructions (such as add, subtract, and compare) in their instruction set. The machine language designed for a processor (CPU), is based on the list of instructions supported by the CPU in its instruction set. Since each processor has a unique instruction set, machine language programs written for one computer will generally not run on another computer, with a different CPU.

CPUs made by different manufacturers have different instruction sets. In fact, different CPU models of the same manufacturer also often have different instruction sets. However, manufacturers tend to group their CPUs into "families", which have similar instruction sets. When a new CPU is developed, it is ensured that its instruction set included all the instructions in the instruction set of its predecessor CPU, plus some new ones. This manufacturing strategy is known as upward compatibility, and the new CPU is said to be upward compatible with its predecessor. This feature allows software written for a computer with a particular CPU, to work on computers with newer processor of the same family. In turn, it allows the users of these computer systems to easily upgrade their system, without worrying about converting all their existing software.

Registers : As the instructions are interpreted and executed by the CPU, there is a movement of information between the various units of the computer system. In order to handle this process satisfactorily, and to speed up the rate of information transfer, the computer uses a number of special memory units, called registers. These registers are used to hold information on a temporary basis, and are part of the CPU (not main memory).

The length of a register equals the number of bits it can store. Hence, a register that can store 8 bits is normally referred to as an 8–bit register. Most CPUs sold today, have 32–bit or 64–bit registers. The size of the registers is sometimes called the word size. The bigger the word size, the faster the computer can process a set of data. With all other parameters being same, a CPU with 32–bit registers, can process data twice as fast as one with 16–bit registers.

Although, the number of registers varies from computer to computer, there are some registers, which are common to all computers. The functions of these registers are described below.

1. Memory Address Register (MAR) : It holds the address of the active memory location. It is loaded

from the program control register, when an instruction is read from memory.

2. Memory Buffer Register (MBR) : It holds the contents of the memory word read from, or written in memory. An instruction word placed in this register is transferred to the instruction register. A data word placed in this register is accessible for operation with the accumulator register, or for transfer to the I/O register. A word to be stored in a memory location must first be transferred to the MBR, from where it is written in memory.

3. Program Control Register (PCR) : It holds the address of the next instruction to be executed. Normally, the instructions of a program are stored in consecutive memory locations, and executed in sequence, unless a branch instruction is encountered. A branch instruction is an operation, which, calls for a transfer to a nonconsecutive instruction. The address part of a branch instruction is transferred to the PC register, to become the address of the next instruction.

4. Accumulator Register (AR) : It holds the data to be operated upon, the intermediate results and the results of processing. It is used during the execution of most instructions. The results of arithmetic operations are returned to the accumulator register, for transfer to main memory, through the memory buffer register. In many computers, there are more than one accumulator registers.

5. Instruction Register (IR) : It holds the current instruction, which is being executed, as soon as the instruction is stored in this register, the operation part and the address part of the instruction are separated. The address part of the instruction is sent to the MAR, while its operation part is sent to the control unit, where it is decoded and interpreted, and ultimately command signals are generated to carry out the task specified by the instruction.

6. Input/Output Register (I/O) : It is used to communicate with the input/output devices. All input information, such as instructions and data, are transferred to this register by an input device. Similarly, all output information, to be transferred to an output device, are found in this register.

Table : Functions of various registers

S.No.	Name of register	Function
1.	Memory Address (MAR)	Holds the address of the active memory location.
2.	Memory Buffer (MBR)	Holds information on its way to and from memory
3.	Program Control (PC)	Holds the address of the next instruction to be executed
4.	Accumulator Register (AR)	Accumulator results and data to be operated upon
5.	Instruction Register (IR)	Holds an instruction, which it is being executed
6.	Input/Output (I/O)	Communicate with the I/O device

The execution of an instruction by the CPU, during program execution, normally involves the following steps:

1. The control unit takes the address of the next program instruction to be executed from the program control register, and reads the instruction from the corresponding memory address, into the instruction register of the control unit.
2. The control unit then sends the operation part and the address part of the instruction, to the decoder and the memory address register, respectively.
3. The decoder interprets the instruction, and accordingly the control unit sends signals to the appropriate unit, which needs to be involved in carrying out the task specified in the instruction. For example, if it is an arithmetic or logic operation, the signal is sent to the ALU. In this case, the control unit also ensures that the data corresponding to the address part of the instruction is loaded in a suitable register in the ALU, before the signal is sent to the ALU. The ALU performs the necessary operation on the data, and signals the control unit as soon as it has finished.
4. As each instruction is executed, the address of the next instruction to be executed is automatically loaded into the program control register, and steps 1 to 4 are repeated.

❑❑❑