1. On which model is based the basic architecture of a digital computer?
**Ans.** The basic architecture of a digital computer is based on Von Neumann model.

2. What is meant by distributed processing?
**Ans.** Distributed processing involves the use of several microprocessors in a single computer system. For example, for such a system, the first microprocessor may control keyboard activities, the second controls storage devices like disk drives, the third controls input/output operations, while the fourth may act as the main system processor.

3. When was the first microprocessor developed?
**Ans.** The first microprocessor was developed by BUSICOM of Japan and INTEL of USA in the year 1971.

4. What is a microprocessor?
**Ans.** A microprocessor may be thought of as a silicon chip around which a microcomputer is built.

5. What is the technology used in microprocessors?
**Ans.** NMOS technology is used in microprocessors.

6. What are the three main units of a digital computer?
**Ans.** The three main units of a digital computer are: the central processing unit (CPU), the memory unit and the input/output devices.

7. How does the microprocessor communicate with the memory and input/output devices?
**Ans.** The microprocessor communicates with the memory and the Input/Output devices via the three buses, viz., data bus, address bus and control bus.

8. What are the different jobs that the CPU is expected to do at any given point of time?
**Ans.** The CPU may perform a memory read or write operation, an I/O read or write operation or an internal activity.

9. What is a mnemonic?
**Ans.** It is very difficult to understand a program if it is written in either binary or hex code. Thus the manufacturers have devised a symbolic code for each instruction, called a mnemonic.
Examples of mnemonics are:
INR A, ADD M, etc.

10. What is machine language programming?
**Ans.** Programming a computer by utilising hex or binary code is known as machine language programming.

11. What is meant by assembly language programming?
**Ans.** Programming a microcomputer by writing mnemonics is known as assembly language programming.

12. What are meant by low level and high level languages?
**Ans.** Programming languages that are machine dependent are called low level languages. For example, assembly language is a low level language.

On the other hand, programming languages that are machine independent are called high level languages. Examples are BASIC, FORTRAN, C, ALGOL, COBOL, etc.

13. What is meant by ‘word length’ of a computer?
**Ans.** The number of bits that a computer recognises and can process at a time is known as its ‘word length’.

14. What is meant by instruction?
**Ans.** An instruction is a command which asks the microprocessor to perform a specific task or job.

15. How many different instructions μP 8085 has? What is an instruction set?
**Ans.** 8085 microprocessor has a total of 74 different instructions for performing different operations or tasks.

The entire different instructions that a particular microprocessor can handle is called its instruction set.

16. What an instruction consists of?
**Ans.** An instruction consists of an operation code (called ‘opcode’) and the address of the data (called ‘operand’), on which the opcode operates.

<table>
<thead>
<tr>
<th>Operation code (or opcode)</th>
<th>Address of data (or operand)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field 1</td>
<td>Field 2</td>
</tr>
</tbody>
</table>

17. Give one example each of the different types of instructions.
**Ans.** Instructions can be of (i) direct (ii) immediate (iii) implicit type. Examples of each type follows:

- (i) direct type : LDA 4000
- (ii) immediate type : MVIA, 1F
- (iii) implicit type : ADD C

18. What language a microprocessor understands?
**Ans.** Microprocessor understands only binary language.

19. How the mnemonics written in assembly language are translated into binary?
**Ans.** The translation from assembly language (i.e., mnemonics) into binary is done either manually (known as hand (or manual) assembly) or by a program called an assembler.
Thus an assembler can be thought of as a program which translates the mnemonics entered by an ASCII keyboard into its equivalent binary code, which is the only one understood by a microprocessor.

20. **How an assembler translates programs written in mnemonic form to binary?**

*Ans.* An assembler has a ‘translation dictionary’, which is stored in its memory. Mnemonics entered via keyboard is compared with this dictionary, which then retrieves its binary equivalent from the same place (dictionary).

21. **What are the types of mnemonics possible?**

*Ans.* Mnemonics can be of two types—alphabetical or alphanumerical.

Example of first type is ADD D whereas, MVI A, 0F is an example of the second category.

22. **What are source codes and object codes?**

*Ans.* High level languages (like, BASIC, COBOL, ALGOL, etc.) are called source codes, whereas binary language is called object code.

23. **How are high level languages converted into binary?**

*Ans.* The high level languages are converted into their corresponding binary by means of another program, called either a compiler or an interpreter.

Compilers are generally used for FORTAN, PASCAL, COBOL, etc. whereas interpreters are generally used in BASIC. M-BASIC is a common example of an interpreter for BASIC language.

24. **What is a ‘statement’?**

*Ans.* Instructions written in high level languages are known as statements.

25. **Write down the difference between a compiler and an interpreter.**

*Ans.* Difference between a compiler and an interpreter lies in the generation of the machine code or object code.

A compiler reads the entire program first and then generates the corresponding object code.

Whereas, an interpreter reads an instruction at a time, produces the corresponding object code and executes the same before it starts reading the next instruction. A program from a compiler runs some 5 to 25 times faster than a program from an interpreter.

26. **Differentiate between a compiler/interpreter and an assembler.**

*Ans.*

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Compiler/Interpreter</th>
<th>Assembler</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Debugging easier</td>
<td>Debugging relatively tougher</td>
</tr>
<tr>
<td>2.</td>
<td>Less efficient</td>
<td>More efficient</td>
</tr>
<tr>
<td>3.</td>
<td>Requires large memory space</td>
<td>Requires less memory space</td>
</tr>
</tbody>
</table>

As an example, for gadget controls and traffic signals, assembly language programming is done because programs in such cases are compact and not lengthy. But for cases where large program lengths are a must, compilers/interpreters are used. For such a case, although a large memory space is required, the advantage lies in very quick debugging.

27. **What are the names given to instructions written in high and low level languages?**

*Ans.* The instructions written in high level languages are known as ‘statements’ whereas those written in low level languages are called ‘mnemonics’.
28. What is another name of a microprocessor?
Ans. A microprocessor is also called the CPU, the central processing unit.

29. What is a microcomputer?
Ans. A microcomputer is a system which is capable of processing a stream of input informations and will generate a stream of output informations taking the help of a program which is stored in the memory of the system.

30. What are the jobs that a microcomputer is capable of doing? How does it do the jobs?
Ans. A microcomputer is capable of doing the following jobs:
(a) receiving input (in the form of data or instruction).
(b) performing computations, both arithmetic and logical.
(c) storing data and instructions.
(d) displaying the results of any computations.
(e) controlling all the devices that perform the above mentioned four tasks, either directly or indirectly.

The first task is done by an input device, whereas the second job is done by the arithmetic logic unit (ALU). The third task is done by the memory unit while an output device does the fourth job.

The fifth job is executed by the timing and control (T&C) unit.

31. What is a bus?
Ans. A bus is a bunch of wires through which data or address or control signals flow.

32. What are the different buses and what jobs they do in a microprocessor?
Ans. The different buses in a microprocessor are the data bus (DB), address bus (AB) and the control bus (CB). Data flow through the DB, while address comes out of the AB and CB controls the activities of the microcomputer system at any instant of time.

33. Why are the different buses buffered?
Ans. A buffer enhances the current or power driving capability of a pin of an IC.

34. In how many ways computers are divided?
Ans. Computers are divided into three categories as per the superiority and number of microprocessors used. These are:
- Micro computers
- Mini computers
- Mainframe computers

35. Distinguish between the three types of computers.
Ans. A microcomputer is a small computer containing only a single central processing unit (CPU). Their word length varies between 8 and 32 bits and used in small industrial and process control systems. The storage capacity and speed requirements of microcomputers are moderate.

Mini computers are having more storage capacity and more speed than micro computers. Mini computers are used in research, data processing, scientific calculations etc.

Mainframe computers are designed to work at very high speed and they have very high storage capacity. Their word length is typically 64-bits. These are used for research, data processing, graphic applications, etc.
36. **In how many ways computer softwares are categorised?**

**Ans.** Computer softwares are divided into two broad categories—system software and user software.

37. **Explain the two types of softwares.**

**Ans.** System software is a collection of programs used for creation, preparation and execution of other programs. Editors, assemblers, loaders, linkers, compilers, interpreters, debuggers and OS (operating system) are included in system software.

User software are softwares developed by various users.

38. **Draw the software hierarchy of a microcomputer system.**

**Ans.** The software hierarchy is shown below:

![Software Hierarchy Diagram](image)

**Fig. 1.1:** Software hierarchy of a microcomputer system

39. **What is an editor?**

**Ans.** An editor is a program and is used for creation and modification of source programs or text. The editor program includes commands which can delete, insert or change lines/characters.

An editor stores in ASCII form in successive RAM locations of the typed letters/numbers. This then can be saved by the SAVE command on a floppy or hard disk. The editor can load the program on RAM later for corrections/alterations, if necessary.

40. **What is an OS (operating system) and what are its functions?**

**Ans.** An operating system provides an interface between the end user and the machine. It performs the function of resource management. By 'resource' is meant a microprocessor, memory or an I/O device.

An OS is a collection of a set of programs that manages machine functions under varied conditions.

The different functions performed by the OS include an efficient or effective way of sharing memory, I/Os and the CPU amongst different users. Today, operating systems are DOS, UNIX, WINDOWS, etc.

41. **What are the different types of assemblers used?**

**Ans.** The different assemblers used are: (a) Macro Assembler, (b) Cross Assembler, (c) Meta Assembler.

42. **What is a linker?**

**Ans.** A linker is a program that links several small object files to produce one large object file.
A large program is usually divided into several small programs. They are written separately, tested and debugged. The large program is then produced by linking these debugged programmes.

A linker has a link file and a link map. The former contains the binary codes of all the modules which have been combined while the link map stores in it the addresses of all the link files. A linker assigns relative addressing instead of absolute addressing which helps in putting a linker program anywhere in the memory.

43. What is a locator?
Ans. A locator is a program which is used to assign specific address when object code is to be loaded into the memory.

44. What are the different assembly languages used for 8085 microprocessor?
Ans. Two different assembly languages are used for 8085 microprocessor—one is used by the manufacturer INTEL and the other defined in IUL 77.

45. What is a coprocessor?
Ans. A coprocessor or an arithmetic coprocessor, as it is so called, performs operations like exponentiation, trigonometric functions, etc. To implement these functions in CPU hardware is a costly one and the software implementation of the above slows down the processor. A coprocessor overcomes these difficulties.

The coprocessor has its own instruction set. The CPU and the coprocessor execute their respective instructions from the same program.

The instructions belonging to the coprocessor are fetched and decoded by the CPU but executed by the coprocessor.

46. Draw a typical coprocessor configuration and discuss the same.
Ans. A typical coprocessor configuration looks like as follows:

![Fig. 1.2: A typical coprocessor configuration](image-url)
The CPU and the coprocessor share the same clock, bus control logic and also share the entire memory and the I/O subsystem. In normal configuration, CPU is the master and the coprocessor is the slave.

The CPU treats the coprocessor as an external memory in the sense that the CPU can read from/write into the coprocessor registers in a manner alike that of an external memory. The CPU and the coprocessor executes their instructions independent of each other. When the coprocessor executes instructions, it indicates the CPU about its executions via the ‘BUSY’ signal.

47. What is a coprocessor trap?
Ans. When a coprocessor is not connected in the system and the functions of the coprocessor is done by the CPU by writing coprocessor instructions from a predetermined location in the form of a software routine and invoked by interrupt, it is known as ‘coprocessor trap’.

48. Explain the three fields contained in a coprocessor instruction.
Ans. There are three fields in a coprocessor instruction—these are code, address and opcode. The code field distinguishes between the coprocessor and CPU instructions. The address field indicates the address of a particular coprocessor. This is required when several coprocessors are used in the system. The opcode field indicates the type of operation to be executed by the processor.

49. What is a debugger?
Ans. A debugger is a program that debugs an object code program by placing it into the system memory and subsequently executing the same.

50. How does a debugger help in debugging a program?
Ans. A debugger is a software tool which isolates the problems/drawbacks in a programmer’s program.

A debugger helps in debugging a program in the following ways:
(a) A debugger helps in checking the contents of memory locations and various registers and also alter the same if required and rerun the program to check the correctness of the modified program.
(b) Program execution can be stopped after each instruction–hence step by step checking is possible with a debugger.
(c) A debugger can set a breakpoint at any place in a program.

A program then can run up to the breakpoint address and not beyond. Thus any fault in the program up to the breakpoint address can be checked by having a look at the various registers and memory contents. If no fault occurs the breakpoint is set at a latter address in the program and the debugger can again be rerun to check for the correctness of the program. This way the whole program can be corrected by judiciously inserting breakpoints in a program.

51. What is meant by the term ‘word’?
Ans. A collection of bits is called a word. A word does not have a fixed number of bits—unlike the case of byte (= 8 byte) or a nibble (= 4 bytes).

For different microprocessors, the word length varies. For example, for 8-bit microprocessors, the word length is 8-bits while that for 32-bit microprocessors, the word length is 32-bits.

A word is expressed usually in multiples of 2, but no value is excluded. That is, we can have a 19-bit word or 37-bit word, etc.
52. What is meant by the term ‘long word’?
Ans. The term ‘long word’ means a group of bits twice the normal length of the microprocessor. Hence, for a 16-bits microprocessor like 8086, a long word means a group consisting of 32-bits.

53. Distinguish between KB, MB, GB, TB and PB.
Ans. These stand for Kilobyte, Megabyte, Gigabyte, Terabyte and Petabyte respectively.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 KB</td>
<td>1024 bytes = $2^{10}$ bytes</td>
</tr>
<tr>
<td>1 MB</td>
<td>1 Kilo KB = $2^{20}$ bytes</td>
</tr>
<tr>
<td>1 GB</td>
<td>1 Kilo MB = $2^{30}$ bytes</td>
</tr>
<tr>
<td>1 TB</td>
<td>1 Kilo GB = $2^{40}$ bytes</td>
</tr>
<tr>
<td>1 PB</td>
<td>1 Kilo TB = $2^{50}$ bytes</td>
</tr>
</tbody>
</table>

54. Compare signed magnitude number and complementary numbers.
Ans. The comparison is made on the basis of an 8-bit word.

In signed magnitude number system, MSB represents the sign of the number with 1 and 0 representing minus and plus, respectively. This number system has the following drawbacks:

- Using the MSB for sign bit leaves 7-bits for data which can range from 0 to $2^{7}_{10} (=1111111_{2})$ while 8-bits can manage numbers from 0 to $2^{8}_{10} (=11111111_{2})$.
- Addition of two signed numbers does not give the correct result always. An example follows:

Two decimal numbers $+127_{10}$ and $+8_{10}$ are to be added via signed number system.

$+127_{10}$ : The Signed no. is $= 01111111$

$+8_{10}$ : The Signed no. is $= 00001000$

\[ (+) \]

This represents the sign of the number. Here it is negative.
Thus the result of addition of the two numbers is $-07_{10}$, but it should have been $+135_{10}$.

Use of complementary numbers (usually 2's complement of a binary number is used) gives rise to the following advantages:
- For an 8-bit data, the data values can vary from 0 to $2^{8}_{10} (=11111111_{2})$
- Addition and subtraction can be implemented with almost same circuitry.

55. What is meant by normalising a number?
Ans. When a number 4751 is converted to $4.751 \times 10^{3}$, the latter is called the normalised version of the former.

Likewise, in binary, a number 1011 1101 can be normalised to $1.0111101 \times 2^{8}$. Another binary number $0.0001101$ is normalised to $1.101 \times 2^{-4}$.

If a given binary number is in a form $1.0101101$, then it is already in the normalised form.

56. Name the different parts of a normalised number.
Ans. Let a binary number is $1.011101 \times 10^{11}$. Then

\[
\begin{align*}
\text{Mantissa} & = 1.011101 \\
\text{Radix} & = 10 \\
\text{Exponent} & = 11
\end{align*}
\]
Hence, a normalised binary number consists of mantissa, radix and exponent.

57. **Discuss the utility of floating point numbers.**

**Ans.** Floating point numbers are useful when the system has either to handle extremely large numbers (as in Astronomy) or very small ones (as in Nuclear Physics). For floating point numbers, the binary number starts with a 1, followed by the binary point. An 8-bit binary mantissa can assume extreme values as 1.0000000 to 1.1111111.

While storing such floating point numbers, the mandatory 1 followed by the binary point can safely be ignored—thereby saving previous storage space. It is assumed that the 1 and the binary point are just present. Thus the above can now be stored as 0000000 to 1111111.

For floating point numbers, a 32-bit storage area is used. Many variants are there to fill in these 32-bits, but the following figure is the one that is most sought after.

![Figure 1.3: Floating point numbers—32-bit storage area](image)

Bit 0 is the sign bit for the mantissa. Bits 1 to 23 hold the mantissa while bits 24 to 31 hold the exponent of the number. The eight bits (bits 24 to 31) are used to hold the numbers from −127 to +128 in 2’s complement form or Excess −127 form. In Excess −127 notation, 127 is added to the original exponent value (in decimal)—this is done to get rid of the negative exponent values. Thus this modified exponent value (positive) is stored in bits 24 to 31 after it is converted into binary.

58. **Discuss speed, size and accuracy while handling floating point numbers.**

**Ans.** Floating point operations that can be carried out per second is a measure of the speed—abbreviated as FLOPS.

Accuracy is dependent on the number of mantissa bits. More number of mantissa bits result in more accuracy. Bits 1 to 23, meant for mantissa, can have a maximum binary value of 1.11111111111111111111. The 1’s, starting from the binary point on the right, have their decimal values +0.5, +0.25, +0.125, +0.0625, etc. Thus, when the 24-bits are converted into decimal, it will approach a value of 2 — but will never equal it. If more number of bits are accommodated in mantissa (i.e., mantissa will now have more than the normal 24-bits), then the converted decimal equivalent will be having a value which will be even more closer to 2. Thus increasing number of mantissa bits result in more **accuracy**.

Size is dependent on the number of exponent bits. More number of exponent bits results in greater size of the number that can be handled. Bits 24 to 31, meant for exponent, has eight bits ranging from −127 to +128. Thus the maximum number that can be handled is \(1 \times 2^{128} = 3.4 \times 10^{38}\) roughly. This value would have been more if in case more number of bits are accommodated into exponent bits. Thus increasing number of exponent bits results in greater **size** of the number that can be handled. In conclusion, when more accuracy is demanded, number of mantissa bits are to be increased (instead of the normal 24-bits) and in cases of higher size of data handling, number of exponent bits are to be increased (instead of the normal 8-bits) within the 32-bit storage space.

59. **Explain single and double precision.**

**Ans.** Single and double precision mean 32-bits and 64-bits storage areas respectively. The extra storage space made available from single to double precision—utilised for
accommodating more number of mantissa bits—would result in much more accuracy from the system.

60. **List the different generation languages.**

**Ans.** These are first, second, third and fourth generation languages. Machine code belongs to first generation, while Assembly Language belongs to second generation language. First and second generation languages are known as low level languages. Third generation languages include FORTRAN, BASIC, COBOL, PASCAL, C, C++, JAVA while fourth generation languages are LISP, APL, PROLOG, etc.

Fig. 1.4 shows pictorially different generation languages.

![Diagram showing different generations of programming languages]

**Fig. 1.4:** The different languages