# BPSC TRE 4.0
# Computer Science

Higher Secondary (Class 11–12) Teacher

# Preface

Dear teacher-to-be,

When BPSC announced TRE 4.0, the Computer Science paper came with a syllabus of one line: the questions will be related to SCERT and NCERT, at the level of the qualification needed for the post. Nobody can plan three months of study from one line. So this book began with a count.

All 234 Computer Science questions of TRE 1.0, 2.0 and 3.0 were sorted, one by one, under the topics they belong to. The count decided the shape of the book. Six areas — digital logic, computer organisation, computer networks, C and C++, linear data structures and operating systems — account for more than half of those questions, so they get the most pages. Python has not yet been asked, but it is the language of the NCERT books, so it is covered in full; TRE 4.0 may well ask it. You will find the count for each chapter in its Exam Radar box, and for the whole paper in "Exam at a Glance".

I have tried to keep three promises in these pages.

- **Every program was run.** The output printed under a program is what the computer printed, not what someone expected it to print.
- **Every numerical was worked twice,** once by hand and once by a program written to check it.
- **Every question is new.** None is copied from a guide or from a past paper, so the practice sets and the mock tests test your understanding and not your memory.

The book is written in plain English, in short sentences. The important terms are also given in Hindi, in the words that BPSC itself uses in its papers, so that nothing surprises you in the examination hall.

A word about what this book does not cover. It prepares you for Part III of the paper, the subject part of 80 questions. The language part and the general studies part need their own preparation.

You are preparing to teach, and not only to pass. I have tried to explain each topic the way a good teacher explains it in class: the idea first, then an example, then the mistake to avoid. If this book helps you to clear the examination, and then helps you a little in your own classroom, it will have done its work.

If you find an error, please write to us. A book of this kind becomes better with every reader.

With best wishes for your success,

**Arun Jha**

# How to Use This Book

## What every chapter contains

All 22 chapters are built in the same way, so that you always know where to look.

| Part of the chapter | What it is for |
|---|---|
| Learning Objectives | What you should be able to do after the chapter. Read them again at the end and check yourself. |
| Exam Radar | How many of the 234 questions of TRE 1.0 to 3.0 came from this chapter, and what was asked. It tells you how much time the chapter deserves. |
| Theory | Short paragraphs, tables and figures. Terms in **bold** are the ones to remember; they are listed in the Index. |
| Worked Example | A problem solved step by step. Cover the solution and try it yourself first. |
| Exam Trap | A mistake that is easy to make, or a point on which BPSC's key is known. |
| Teacher's Tip | A way to teach the idea in a classroom. |
| Key Terms | The main terms in English and in the Hindi of BPSC's papers. |
| Quick Revision | The whole chapter in ten or twelve lines. |
| Practice Set | Original questions at three levels, then the answer key and an explanation of every answer. |

This is how the two boxes you will meet most often look:

> **Remember**
> A point worth learning by heart is set in a box like this one.

> **Exam Trap**
> A common mistake, and the way to avoid it, is set in a box like this one.

## The three levels of questions

| Level | Name | What it tests | Share |
|---|---|---|---|
| 1 | Foundation | Direct recall of a fact or a definition | about 30% |
| 2 | Exam Standard | The level of BPSC's own questions: statements to judge, assertion and reason, matching, short calculations, the output of a program | about 50% |
| 3 | Challenger | Two-step problems and the hardest questions likely in a paper set at graduate level | about 20% |

If you are short of time, do Levels 1 and 2 of every chapter before Level 3 of any.

## How to study a chapter

1. Read the Exam Radar and the Learning Objectives.
2. Read the theory with a pen in hand. Work every example on paper before you read its solution, and type and run the programs if you have a computer.
3. Do the practice set without looking back. Mark your answers, then read the explanations of the questions you got wrong **and** of those you guessed.
4. Two or three days later, read the Quick Revision box and do again only the questions you got wrong.

## The mock tests

The five mock tests have the size and the rules of Part III of the real paper: 80 questions, one mark each, one-third of a mark deducted for a wrong answer. Each question has the fifth option "(E) Not attempting", as in TRE 4.0. In the real examination a question left with no circle darkened also loses one-third of a mark, so practise marking (E) whenever you decide to skip.

Take each test in one sitting of 80 minutes, on the answer sheet printed after it. Then fill in the score table on the key page; it shows your marks area by area and tells you which chapters to read again. The tests become harder from 1 to 5.

## A plan for 90 days

The plan assumes about three hours a day for this paper: two for reading and one for questions. Chapters of high priority get more days.

| Days | Chapters | What to finish |
|---|---|---|
| 1–7 | 1, 2 | Fundamentals; number systems and codes |
| 8–14 | 3 | Boolean algebra and digital logic, with every K-map example |

| Days | Chapters | What to finish |
|---|---|---|
| 15–21 | 4, 5 | Computer organisation; operating systems |
| 22–28 | 6, 7, 8 | Problem solving; Python I and II |
| 29–35 | 9, 10, 11 | Object-oriented programming; C and C++; Java |
| 36–42 | 12, 13 | Algorithm analysis; linear data structures |
| 43–49 | 14, 15 | Trees and graphs; searching, sorting and hashing |
| 50–56 | 16, 17 | DBMS concepts; SQL |
| 57–63 | 18, 19 | Computer networks; Internet, web and multimedia |
| 64–70 | 20, 21, 22 | Software engineering; cyber security; emerging trends |
| 71–77 | Revision | Quick Revision boxes of Chapters 3, 4, 5, 10, 13 and 18; Mock Tests 1 and 2 |
| 78–84 | Revision | Weak areas shown by the score tables; Mock Tests 3 and 4 |
| 85–90 | Last week | Mock Test 5; the 250 one-liners, twice |

If you have less than 90 days, keep the order and shorten every block in the same proportion. Do not drop the mock tests.

# Exam at a Glance

The pattern below is the one notified by BPSC for TRE 4.0. Always check the Commission's website, bpsc.bihar.gov.in, for later notices.

## The paper for Class 11–12 teachers

| Part | Subject | Questions | Marks | Counts for merit? |
|---|---|---|---|---|
| I | Language: English, and Hindi, Urdu or Bangla | 30 | 30 | No. It is a qualifying part; at least 30% is needed. |
| II | General Studies: elementary mathematics, general awareness, general science, Indian national movement, geography | 40 | 40 | Yes |
| III | **Subject: Computer Science** | **80** | **80** | Yes |
| | **Total** | **150** | **150** | Time: 2 hours 30 minutes |

| Rule | What BPSC has laid down |
|---|---|
| Type of question | Objective, multiple choice, with five options (A) to (E) |
| Correct answer | One of (A) to (D) |
| Option (E) | "Not attempting". A candidate who does not want to answer a question must mark (E). |
| Wrong answer | 1/3 mark is deducted |
| No option marked | 1/3 mark is deducted, as for a wrong answer |
| Minimum qualifying marks | General 40%; BC 36.5%; EBC 34%; SC, ST, women and divyang 32% |
| Merit list | The total of Part II and Part III. In a tie, higher marks in Part III come first. |
| Syllabus of Part III | "Related to SCERT/NCERT", at the level of the minimum qualification for the post. No topic-wise syllabus is given. |
| Attempts | At most 5 |
| Re-evaluation | Not allowed |

## What BPSC has asked so far

The chart counts the Computer Science questions of TRE 1.0 (2023), TRE 2.0 (2023) and TRE 3.0 (2024) under the chapters of this book. “Priority” is the weight given to the chapter in this book. Python and searching and sorting have not been asked yet, but they are the core of the NCERT course.

| Ch | Chapter | Qs | Share of the 234 questions | Priority |
|---|---|---|---|---|
| 1 | Computer Fundamentals | 12 |  | Medium |
| 2 | Number Systems and Data Representation | 9 |  | Medium |
| 3 | Boolean Algebra and Digital Logic | 26 |  | High |
| 4 | Computer Organisation and Architecture | 25 |  | High |
| 5 | Operating Systems | 18 |  | High |
| 6 | Problem Solving and Programming Concepts | 0 | – | Low |
| 7 | Python I | 0 | – | Medium (NCERT) |
| 8 | Python II | 0 | – | Medium (NCERT) |
| 9 | Object-Oriented Programming | 14 |  | High |
| 10 | C and C++ | 19 |  | High |
| 11 | Java Basics | 2 |  | Low |
| 12 | Algorithm Analysis and Theory of Computation | 5 |  | Medium |
| 13 | Linear Data Structures | 20 |  | High |
| 14 | Trees and Graphs | 10 |  | Medium |
| 15 | Searching, Sorting and Hashing | 2 |  | Medium (NCERT) |
| 16 | DBMS Concepts | 12 |  | Medium |
| 17 | SQL and Python–MySQL Connectivity | 15 |  | High |
| 18 | Computer Networks | 24 |  | High |
| 19 | Internet, Web and Multimedia | 8 |  | Medium |
| 20 | Software Engineering | 4 |  | Low |
| 21 | Cyber Security, Law and Ethics | 6 |  | Medium |
| 22 | Emerging Trends | 3 |  | Low |

# Contents


**Preface** vi

**How to Use This Book** vii

**Exam at a Glance** x

**Part A – Computer Systems & Organisation** 1

**1 Computer Fundamentals** 2

1.1 Evolution and generations of computers . . . 2
1.2 Types of computers . . . 3
1.3 Components of a computer system . . . 4
1.4 Input, output and storage devices . . . 5
1.5 Hardware and software . . . 5
1.6 Programming languages and translators . . . 6
1.7 Units of memory . . . 7

**2 Number Systems and Data Representation** 14

2.1 Number systems and conversions . . . 14
2.2 Binary arithmetic . . . 16
2.3 Signed numbers . . . 17
2.4 Floating-point numbers . . . 18
2.5 Binary codes . . . 19
2.6 Error-detecting and error-correcting codes . . . 19
2.7 Character codes . . . 20

**3 Boolean Algebra and Digital Logic** 25

3.1 Logic gates and truth tables . . . 25
3.2 Laws of Boolean algebra . . . 27
3.3 Canonical forms . . . 28
3.4 Simplification with Karnaugh maps . . . 29
3.5 Combinational circuits . . . 31
3.6 Sequential circuits . . . 32

**4 Computer Organisation and Architecture** 40

4.1 The CPU and its registers . . . 40
4.2 Instruction cycle and instruction formats . . . 41
4.3 Addressing modes . . . 42
4.4 Computer arithmetic: Booth's algorithm . . . 43
4.5 Memory . . . 44
4.6 Secondary storage . . . 47
4.7 Input–output organisation . . . 47
4.8 Buses . . . 48
4.9 Microprocessors and microcontrollers . . . 48

**5 Operating Systems** 56

5.1 Functions and types of operating systems . . . 56
5.2 Processes and threads . . . 57
5.3 CPU scheduling . . . 58
5.4 Synchronisation and deadlock . . . 60
5.5 Memory management . . . 61
5.6 Virtual memory . . . 62
5.7 File systems and disk scheduling . . . 64
5.8 Windows and Linux essentials . . . 65

**Part B – Programming** 73

**6 Problem Solving and Programming Concepts** 74

6.1 Steps in problem solving . . . 74
6.2 Algorithms, flowcharts and pseudocode . . . 74
6.3 Decomposition . . . 76
6.4 Types of errors and debugging . . . 76
6.5 Programming paradigms . . . 77

**7 Python I: Fundamentals and Built-in Data Types** 81

7.1 Tokens, variables and data types . . . 81
7.2 Mutable and immutable types . . . 82
7.3 Operators and precedence . . . 83
7.4 Control flow . . . 84
7.5 Strings . . . 86
7.6 Lists . . . 87

7.7 Tuples . . . 88
7.8 Dictionaries . . . 88
7.9 Sets . . . 89
7.10 Built-in functions and standard modules . . . 90

**8 Python II: Functions, Exceptions and Files 97**

8.1 Functions . . . 97
8.2 Scope of variables and the LEGB rule . . . 98
8.3 Recursion . . . 99
8.4 Modules and packages . . . 100
8.5 Exception handling . . . 100
8.6 Text files . . . 101
8.7 Binary files and the pickle module . . . 103
8.8 CSV files . . . 104

**9 Object-Oriented Programming 111**

9.1 Concepts of object-oriented programming . . . 111
9.2 Classes and objects in C++ . . . 112
9.3 Access specifiers . . . 113
9.4 Constructors and destructors . . . 113
9.5 Inheritance . . . 114
9.6 Polymorphism . . . 115
9.7 Abstraction and abstract classes . . . 116
9.8 Object-oriented programming in Python . . . 117

**10 C and C++ 125**

10.1 Structure of a program and compilation . . . 125
10.2 Data types, variables and operators . . . 126
10.3 Control structures . . . 128
10.4 Functions . . . 129
10.5 Arrays and strings . . . 129
10.6 Pointers and dynamic memory . . . 130
10.7 Structures and unions . . . 131
10.8 File handling . . . 132
10.9 Output-prediction techniques . . . 133

**11 Java Basics 141**

11.1 Bytecode, JVM, JRE and JDK . . . 141
11.2 Classes, objects and access levels . . . 142
11.3 Inheritance and interfaces . . . 142
11.4 Exceptions . . . 143
11.5 Java and C++ compared . . . 144

**Part C – Data Structures & Algorithms 149**

**12 Algorithm Analysis and Theory of Computation 150**

12.1 Time and space complexity . . . 150
12.2 Recurrences . . . 151
12.3 Design paradigms . . . 152
12.4 P, NP and NP-complete problems . . . 153
12.5 Automata in brief . . . 154

**13 Linear Data Structures 159**

13.1 Arrays and sparse matrices . . . 159
13.2 Stacks . . . 160
13.3 Infix, prefix and postfix expressions . . . 161
13.4 Queues . . . 163
13.5 Linked lists . . . 165

**14 Trees and Graphs 173**

14.1 Tree terms and binary trees . . . 173
14.2 Traversals . . . 174
14.3 Binary search trees . . . 176
14.4 AVL trees and heaps . . . 177
14.5 Graphs and their representation . . . 178
14.6 BFS and DFS . . . 179
14.7 Spanning trees . . . 179

**15 Searching, Sorting and Hashing 187**

15.1 Linear and binary search . . . 187
15.2 Bubble, selection and insertion sort . . . 188
15.3 Merge sort, quick sort and heap sort . . . 190
15.4 Comparison of sorting algorithms . . . 191
15.5 Hashing . . . 191

**Part D – Databases 199**

**16 DBMS Concepts 200**

16.1 File systems, DBMS and data models . . . 200
16.2 The entity–relationship model . . . 201
16.3 The relational model . . . 202
16.4 Keys and integrity constraints . . . 203
16.5 Relational algebra . . . 203
16.6 Functional dependency and normalisation . . . 204
16.7 Transactions and ACID . . . 206
16.8 Distributed databases . . . 206

**17 SQL and Python–MySQL Connectivity 212**

17.1 DDL, DML, DCL and TCL . . . . . 213
17.2 Data types and constraints . . . . . 214
17.3 The SELECT statement . . . . . 215
17.4 Single-row and aggregate functions . . 217
17.5 GROUP BY and HAVING. . . . . 218
17.6 Joins. . . . . 218
17.7 Subqueries, views and the WITH clause . . . . . 219
17.8 Python–MySQL connectivity. . . . . 221

**Part E – Networks & the Web 229**

**18 Computer Networks 230**

18.1 Evolution of networking and data communication . . . . . 230
18.2 Transmission media . . . . . 232
18.3 Network devices . . . . . 232
18.4 Topologies and types of networks . . . 233
18.5 The OSI and TCP/IP models . . . . . 234
18.6 Protocols and port numbers . . . . . 235
18.7 IP addressing . . . . . 236
18.8 Switching, error control and access methods. . . . . 238
18.9 Wireless and mobile communication . 239

**19 Internet, Web and Multimedia 246**

19.1 The World Wide Web, URLs, browsers and servers. . . . . 246
19.2 HTML . . . . . 247
19.3 CSS, JavaScript and scripting . . . . . 249
19.4 XML and cookies . . . . . 249
19.5 E-commerce . . . . . 250
19.6 Multimedia and computer graphics . . 250

**Part F – Software, Security & Society 257**

**20 Software Engineering 258**

20.1 The software development life cycle . . 258
20.2 Requirements . . . . . 259
20.3 Design and UML . . . . . 260
20.4 Testing . . . . . 260
20.5 Maintenance and estimation . . . . . 261

**21 Cyber Security, Law & Ethics 266**

21.1 Malware and attacks . . . . . 266
21.2 Firewalls, encryption and digital signatures . . . . . 267
21.3 Safe practices . . . . . 269
21.4 Digital footprint and netiquette . . . . 269
21.5 Intellectual property, plagiarism and licences . . . . . 269
21.6 Cyber crime and the IT Act, 2000 . . . 270
21.7 E-waste and health . . . . . 271

**22 Emerging Trends 276**

22.1 Artificial intelligence, machine learning and NLP . . . . . 276
22.2 Big data and data handling . . . . . 277
22.3 Cloud computing. . . . . 278
22.4 The Internet of Things . . . . . 279
22.5 Blockchain . . . . . 279
22.6 Immersive experience, robotics and quantum computing . . . . . 280

**Practice – Mock Tests & Revision 285**

**Mock Test 1 286**

**Mock Test 2 299**

**Mock Test 3 312**

**Mock Test 4 325**

**Mock Test 5 338**

**Last-Week Revision: 250 One-Liners 351**

**Glossary: English to Hindi 358**

**Appendices 362**

A. ASCII Codes . . . . . 362
B. Python Quick Reference. . . . . 362
C. SQL Quick Reference . . . . . 365
D. Common Port Numbers. . . . . 365
E. Linux Commands . . . . . 366
F. Full Forms . . . . . 366

**Index 369**

CHAPTER 1

# Computer Fundamentals

**Learning Objectives**

After this chapter you will be able to:

- place the main machines and inventions of computing in order, generation by generation;
- classify computers by size, by purpose and by the kind of data they handle;
- name the units of a computer system and say what each one does;
- sort common devices into input, output and storage, and software into system, application and utility;
- tell an assembler, a compiler and an interpreter apart;
- convert between bits, bytes, KB, MB, GB and TB.

**Exam Radar**

**Priority: Medium.** This chapter gave 12 of the 234 Computer Science questions in TRE 1.0 to 3.0 (5.1%), nine of them in TRE 2.0 alone. They were all one-line recall: the father of the computer, the brain of the computer, assembler, compiler and interpreter, what “tera” means, the device used for video games. Learn the tables in this chapter by heart.

A **computer** is an electronic machine that accepts data as input, processes it by following stored instructions, and gives information as output. The instructions are called a **program**. Raw facts are **data**; processed, meaningful data is **information**.

## 1.1 Evolution and generations of computers

### 1.1.1 Milestones

| Year | Milestone |
|---|---|
| about 1642 | Blaise Pascal builds the Pascaline, a mechanical adding machine |
| 1830s | Charles Babbage designs the Analytical Engine, a general-purpose mechanical computer controlled by punched cards |
| 1843 | Ada Lovelace publishes her notes on the Analytical Engine; she is often called the first programmer |
| 1945 | John von Neumann writes the report that sets out the stored-program idea |
| 1946 | ENIAC, an electronic computer built with vacuum tubes, is unveiled |
| 1947 | Bardeen and Brattain build the first transistor at Bell Labs |
| 1951 | The first UNIVAC I is delivered to the US Census Bureau |
| 1958 | Jack Kilby of Texas Instruments demonstrates the first integrated circuit (IC) |
| 1971 | Intel introduces the first microprocessor, the 4004 |

| | |
|---|---|
| 1981 | IBM introduces its Personal Computer (PC) |
| 1991 | C-DAC builds PARAM 8000, India's first supercomputer |

> **Remember**
> **Charles Babbage** is called the father of the computer. The main raw material of computer chips is **silicon**, a semiconductor. Both were asked in TRE 1.0.

### 1.1.2 The five generations

Each generation is marked by the electronic component used for processing. The periods are approximate; textbooks differ by a few years.

| Generation | Period (about) | Main component | Languages and software | Example |
|---|---|---|---|---|
| First | 1940–1956 | Vacuum tube | Machine language | ENIAC, UNIVAC I |
| Second | 1956–1963 | Transistor | Assembly language; early high-level languages | — |
| Third | 1964–1971 | Integrated circuit (IC) | High-level languages; operating systems | IBM System/360 |
| Fourth | 1971 onwards | Microprocessor (VLSI) | 4GLs, GUI, networks | IBM PC |
| Fifth | Present and future | ULSI, parallel processing | Artificial intelligence, natural language | — |

From one generation to the next, computers became smaller, faster, cheaper and more reliable, and they used less power and gave off less heat.

## 1.2 Types of computers

**By size and power**

| Type | Main features | Typical use |
|---|---|---|
| Microcomputer | One microprocessor; one user at a time. Desktops, laptops, tablets, smartphones. | Homes, schools, offices |
| Minicomputer | Mid-range; serves some tens or hundreds of users | Departments, small organisations |
| Mainframe | Very large storage; serves thousands of users at once | Banking, railway reservation, census |
| Supercomputer | The fastest; thousands of processors working in parallel. Speed is measured in **FLOPS**. | Weather forecasting, research. Cray-1 (1976); PARAM 8000 (India, 1991) |

**By the data handled**

- An **analog** computer measures quantities that change continuously, such as temperature, pressure or speed. A speedometer is an example.

- A **digital** computer counts. It works on discrete values, the binary digits 0 and 1. Almost all present-day computers are digital.
- A **hybrid** computer combines both. Hospital machines that measure a patient's heartbeat (analog) and show it as numbers (digital) are hybrid.

**By purpose**, a computer is **general-purpose** (it can run many kinds of programs) or **special-purpose** (it is built for one task, like the computer inside a washing machine or an ATM).

## 1.3 Components of a computer system

A computer works on the **input–process–output** cycle. In John von Neumann's **stored-program** design, the program and the data are kept in the same memory, and the processor takes the instructions from memory one after another.

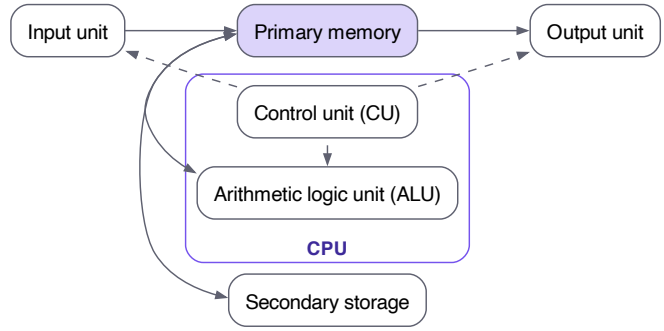



**Figure 1.1.** Block diagram of a computer. Solid arrows carry data; dashed arrows carry control signals.

| Unit | What it does |
|---|---|
| Input unit | Accepts data and instructions and turns them into binary form |
| CPU | The **brain of the computer**. It carries out the instructions. |
| Control unit (CU) | Fetches and decodes each instruction and sends control signals to all other units. It does no calculation itself. |
| Arithmetic logic unit (ALU) | Does arithmetic (+, −, ×, ÷) and logical work (comparisons, AND, OR, NOT) |
| Registers | Very small, very fast stores inside the CPU for the data in use |
| Primary memory | Holds the programs and data in use. RAM is volatile; ROM is not. |
| Secondary storage | Keeps programs and data permanently, even without power |
| Output unit | Turns results from binary form into a form people can use |

> **⚠ Exam Trap**
> BPSC asked for "the three main components of a digital computer system". The keyed answer was **memory, CPU and I/O**. The ALU is a part *of* the CPU, so "ALU, CPU, memory" counts the same unit twice.

## 1.4 Input, output and storage devices

| Input device | Used for |
|---|---|
| Keyboard, mouse, trackball, touchpad | Typing and pointing |
| Joystick | Playing video games; also flight simulators |
| Scanner | Turning a page or photo into an image |
| OMR (optical mark reader) | Reading pencil or pen marks, as on the OMR answer sheets of BPSC |
| OCR (optical character reader) | Turning printed or handwritten text into editable text |
| MICR (magnetic ink character reader) | Reading the numbers printed in magnetic ink on bank cheques |
| Barcode reader, QR scanner | Reading product codes and UPI payment codes |
| Microphone, webcam, biometric sensor | Sound, video, fingerprints (as in Aadhaar verification) |

| Output device | Notes |
|---|---|
| Monitor (LCD, LED) | Soft copy. Its sharpness depends on the resolution, the number of pixels. |
| Printer | Hard copy. **Impact** printers strike a ribbon: dot matrix, daisy wheel, line printer. **Non-impact** printers do not: inkjet, laser (uses toner), thermal. |
| Plotter | Large, precise drawings such as maps and building plans |
| Speaker, projector | Sound; a large picture for a class |

A **touch screen** is both an input and an output device. Printer quality is measured in **dpi** (dots per inch), and speed in characters, lines or pages per minute.

**Storage.** Primary memory (RAM, ROM, cache) is directly used by the CPU; it is fast, costly and small. Secondary storage is slower, cheaper and large: the hard disk and magnetic tape are magnetic, the CD, DVD and Blu-ray disc are optical (read by a laser), and the SSD and pen drive are flash memory with no moving parts. Chapter 4 treats memory in detail.

## 1.5 Hardware and software

**Hardware** is the physical part of a computer, which can be seen and touched. **Software** is the set of programs that makes the hardware work. **Firmware** is software stored permanently in a ROM chip, such as the start-up program of a PC.

| Kind of software | Purpose | Examples |
|---|---|---|
| System software | Runs the computer itself and gives a base for other programs | Operating system (Windows, Linux, Android), language translators, device drivers |
| Utility software | Helps to maintain and protect the system. It is often counted as a part of system software. | Antivirus, disk defragmenter, backup and file-compression tools |
| Application software, general-purpose | Lets the user do a common kind of work | Word processor, spreadsheet, web browser |
| Application software, specific-purpose | Made for one particular task | Railway reservation system, school fee software, payroll |

> **Worked Example 1.1 — Classifying software**
> A school office uses: (i) Windows 11, (ii) an antivirus program, (iii) a fee-collection package, (iv) a spreadsheet, (v) a printer driver. Classify each.
> (i) and (v) are **system software**: the operating system and a device driver. (ii) is **utility software**. (iii) is **specific-purpose application software**, made for one task. (iv) is **general-purpose application software**.

## 1.6 Programming languages and translators

| Level | Language | Features |
|---|---|---|
| Low | Machine language (1GL) | Written in 0s and 1s. The only language the CPU understands. Needs no translator. Depends on the machine. |
| Low | Assembly language (2GL) | Uses short words called mnemonics, such as ADD and MOV. Depends on the machine. Needs an assembler. |
| High | High-level language (3GL) | Close to English: C, C++, Java, Python. Portable across machines. Needs a compiler or an interpreter. |
| High | Fourth-generation language (4GL) | Says *what* is wanted, not *how*: SQL |

A program written by the programmer is **source code**. Its translated machine-language form is **object code**. A **translator** is system software that does this conversion.

| Translator | Input | How it works |
|---|---|---|
| Assembler | Assembly language | Converts each mnemonic into its machine code |
| Compiler | High-level language | Translates the **whole program** at once and lists all its errors together. If there is none, it produces object code, which runs fast and can be run again without the compiler. Used by C and C++. |
| Interpreter | High-level language | Translates and runs **one statement at a time**. It stops at the first error. It makes no object code, so the program is translated on every run and runs more slowly. Used by Python. |

Two more system programs finish the job. A **linker** joins the object code with library routines to make one executable file. A **loader** places the executable file in main memory for running.

> **Worked Example 1.2 — Compiler and interpreter on the same error**
> A program of 80 lines has one syntax error, on line 50. What happens with a compiler, and with an interpreter?
> The compiler scans all 80 lines first. It reports the error on line 50 and makes no object code, so **not a single line runs**. The interpreter runs lines 1 to 49, shows their output, and **stops at line 50**. (This is the textbook model of an interpreter, and the answer expected in examinations. Python checks the whole file for syntax errors before it runs anything; it stops part-way like this only for an error that arises while running.)

> **Teacher's Tip**
> Compare the two translators with two ways of reading a Hindi story to an English-speaking friend. A compiler is like translating the whole story on paper first and then handing it over. An interpreter is like translating aloud, one sentence at a time. Students remember "whole versus line by line" from this.

## 1.7 Units of memory

The smallest unit is the **bit** (binary digit), which is 0 or 1. Four bits make a **nibble** and eight bits make a **byte**. Larger units go up in steps of 1024, which is $2^{10}$.

| Unit | Equal to | In bytes |
|---|---|---|
| 1 kilobyte (KB) | 1024 bytes | $2^{10}$ |
| 1 megabyte (MB) | 1024 KB | $2^{20}$ |
| 1 gigabyte (GB) | 1024 MB | $2^{30}$ |
| 1 terabyte (TB) | 1024 GB | $2^{40}$ |
| 1 petabyte (PB) | 1024 TB | $2^{50}$ |

After the petabyte come the exabyte, zettabyte and yottabyte, each 1024 times the one before.

**Exam Trap**

In everyday science, kilo, mega, giga and tera mean $10^3$, $10^6$, $10^9$ and $10^{12}$. For computer memory, BPSC keys them as powers of two: in TRE 2.0 the answer to "Tera means" was $\mathbf{2^{40}}$. Use powers of two in memory questions unless the question says otherwise.

### Worked Example 1.3 — Working with memory units

(a) How many bits are there in 2 KB? (b) How many photos of 4 MB each fit on a 16 GB pen drive?

(a) 2 KB = 2 × 1024 bytes = 2048 bytes. Each byte has 8 bits, so 2048 × 8 = **16,384 bits**.

(b) 16 GB = 16 × 1024 MB = 16,384 MB. Number of photos = 16,384 ÷ 4 = **4096**.

### Key Terms — English and Hindi

| English | Hindi | English | Hindi |
|---|---|---|---|
| Program | प्रोग्राम | Memory | मेमोरी |
| High-level language | उच्च-स्तरीय भाषा | Machine language | मशीन भाषा |
| Statement | कथन | To compile | संकलित करना |

### Quick Revision

- Father of the computer: Charles Babbage. First programmer: Ada Lovelace. Chips are made of silicon.
- Generations: vacuum tube, transistor, IC, microprocessor, AI.
- ENIAC 1946, transistor 1947, IC 1958, Intel 4004 in 1971, IBM PC 1981, PARAM 8000 in 1991.
- By size: micro, mini, mainframe, super. By data: analog, digital, hybrid.
- Three main components: memory, CPU, I/O. The CPU (brain) has the CU and the ALU.
- OMR reads marks, OCR reads text, MICR reads cheques. The joystick is for games.
- Impact printers: dot matrix, daisy wheel, line. Non-impact: inkjet, laser, thermal.
- System software runs the machine; application software does the user's work; utilities maintain it.
- Assembler: assembly to machine code. Compiler: whole program at once. Interpreter: line by line.
- 1 byte = 8 bits; 1 nibble = 4 bits. KB, MB, GB, TB = $2^{10}$, $2^{20}$, $2^{30}$, $2^{40}$ bytes.

# Practice Set 1

## Level 1 — Foundation

**1.** The Analytical Engine was designed by

(A) Blaise Pascal
(B) Charles Babbage
(C) John von Neumann
(D) Jack Kilby

**2.** The main electronic component of first-generation computers was the

(A) transistor
(B) integrated circuit
(C) vacuum tube
(D) microprocessor

**3.** A nibble is a group of

(A) 4 bits (B) 8 bits
(C) 16 bits (D) 1024 bits

**4.** Which of the following is an input device?

(A) Plotter (B) Speaker
(C) Scanner (D) Projector

**5.** Which translator translates and runs a high-level program one statement at a time?

(A) Compiler (B) Assembler
(C) Linker (D) Interpreter

**6.** Which part of the CPU carries out additions and comparisons?

(A) Control unit
(B) Arithmetic logic unit
(C) Program counter
(D) Cache memory

**7.** Which of the following is system software?

(A) A word processor
(B) A payroll package
(C) An operating system
(D) A web browser

**8.** PARAM 8000, India's first supercomputer, was built by

(A) C-DAC (B) ISRO (C) DRDO (D) BARC

**9.** The numbers printed at the bottom of a bank cheque are read by

(A) OMR
(B) OCR
(C) MICR
(D) a barcode reader

**10.** Which of these units of memory is the largest?

(A) Gigabyte (B) Terabyte
(C) Megabyte (D) Petabyte

**11.** Software that is stored permanently in a ROM chip is called

(A) firmware (B) shareware
(C) freeware (D) groupware

**12.** Which of the following is an impact printer?

(A) Laser printer
(B) Inkjet printer
(C) Thermal printer
(D) Dot matrix printer

## Level 2 — Exam Standard

**13.** Consider these statements about the generations of computers.

1. Second-generation computers used transistors.
2. Integrated circuits were first used in the fourth generation.
3. The microprocessor marks the fourth generation.

Which of the statements are correct?

(A) 1 and 2 only (B) 1 and 3 only
(C) 2 and 3 only (D) 1, 2 and 3

**14.** **Assertion (A):** An interpreted program generally runs more slowly than a compiled one.

**Reason (R):** An interpreter translates the statements again every time the program is run.

(A) Both A and R are true, and R is the correct explanation of A
(B) Both A and R are true, but R is not the correct explanation of A
(C) A is true, but R is false
(D) A is false, but R is true

**15.** Match each device in List-I with its use in List-II.

**List-I:** P. OMR Q. MICR R. Barcode reader S. Plotter

**List-II:** 1. processing bank cheques 2. large engineering drawings 3. marks on answer sheets 4. product labels at a billing counter

(A) P-1, Q-3, R-4, S-2
(B) P-3, Q-1, R-2, S-4
(C) P-4, Q-1, R-3, S-2
(D) P-3, Q-1, R-4, S-2

**16.** Arrange these units from the smallest to the largest: 1. KB 2. TB 3. byte 4. GB 5. MB

(A) 3, 5, 1, 4, 2 (B) 1, 3, 5, 2, 4
(C) 3, 1, 5, 4, 2 (D) 3, 1, 4, 5, 2

**17.** Which of the following is NOT done by the control unit?

(A) Fetching the next instruction
(B) Decoding the instruction
(C) Adding two numbers
(D) Sending control signals to other units

**18.** Arrange these events in order of time, earliest first: 1. Intel 4004 introduced 2. ENIAC unveiled 3. IBM PC introduced 4. First transistor built

(A) 2, 4, 1, 3 (B) 4, 2, 1, 3
(C) 2, 4, 3, 1 (D) 2, 1, 4, 3

**19.** Consider these statements about a compiler.

1. It translates the whole program before any of it runs.
2. It produces object code.
3. It reports each error only when the faulty statement is being run.

Which of the statements are correct?

(A) 1 and 2 only (B) 2 and 3 only
(C) 1 and 3 only (D) 1, 2 and 3

**20.** The system program that joins object code with library routines to make one executable file is the

(A) loader (B) linker
(C) assembler (D) interpreter

**21.** Which of the following pairs is correctly matched?

(A) Joystick — output device
(B) Plotter — input device
(C) Touch screen — input and output device
(D) Scanner — output device

**22.** How many megabytes make one terabyte?

(A) $2^{10}$ (B) $2^{20}$ (C) $2^{30}$ (D) $2^{40}$

**23.** **Assertion (A):** A compiler is system software.

**Reason (R):** A compiler is used by the end user to prepare letters and accounts.

(A) Both A and R are true, and R is the correct explanation of A
(B) Both A and R are true, but R is not the correct explanation of A
(C) A is true, but R is false
(D) A is false, but R is true

**24.** Which of the following is NOT system software?

(A) A device driver
(B) An operating system
(C) An assembler
(D) A spreadsheet

**25.** Consider these statements about the stored-program (von Neumann) design.

1. The program and the data are kept in the same memory.
2. Instructions are taken from memory one after another.
3. It needs one memory for instructions and a separate memory for data.

Which of the statements are correct?

(A) 1 and 2 only (B) 2 and 3 only
(C) 1 and 3 only (D) 1, 2 and 3

**26.** A computer that measures a continuously changing quantity, such as temperature, is

(A) an analog computer
(B) a digital computer
(C) a mainframe computer
(D) a microcomputer

**27.** Which printer is a non-impact printer that uses toner?

(A) Dot matrix printer
(B) Line printer
(C) Daisy wheel printer
(D) Laser printer

**28.** The speed of a supercomputer is usually stated in

(A) dpi (B) FLOPS
(C) bps (D) pixels

**29.** Which of the following is a fourth-generation language?

(A) Assembly language
(B) Machine language
(C) SQL
(D) C

**30.** A language that uses mnemonics such as ADD and MOV is

(A) machine language
(B) assembly language
(C) a high-level language
(D) a fourth-generation language

**31.** Which of the following is utility software?

(A) A compiler
(B) A payroll package
(C) An operating system
(D) A disk defragmenter

**32.** Which type of computer serves thousands of users at the same time, as in banking and railway reservation?

(A) Microcomputer
(B) Minicomputer
(C) Mainframe computer
(D) Analog computer

## Level 3 — Challenger

**33.** How many songs of 8 MB each can be stored on a 32 GB memory card?

(A) 4096 (B) 4000 (C) 2048 (D) 8192

**34.** The number of bits in 4 KB is

(A) 4096 (B) 32,768
(C) 32,000 (D) 8192

**35.** A program of 80 lines has a single syntax error, on line 50. Consider these statements.

1. With a compiler, no line of the program runs.
2. With an interpreter, lines 1 to 49 run before the error is reported.
3. With a compiler, lines 1 to 49 run and then the program stops.

Which of the statements are correct?

(A) 1 only (B) 2 only
(C) 2 and 3 only (D) 1 and 2 only

**36.** Consider these statements.

1. ROM is volatile memory.
2. Cache memory is faster than RAM.
3. Secondary storage keeps data without power.
4. Registers are inside the CPU.

Which of the statements are correct?

(A) 2, 3 and 4 only
(B) 1, 2 and 3 only
(C) 3 and 4 only
(D) 1, 2, 3 and 4

**37.** A CD holds 700 MB. The smallest number of such CDs needed to store 4 GB of data is

(A) 4 (B) 5 (C) 7 (D) 6

**38.** Match each generation in List-I with its main component in List-II.

**List-I:** P. First Q. Second R. Third S. Fourth

**List-II:** 1. Microprocessor 2. Vacuum tube 3. Integrated circuit 4. Transistor

(A) P-2, Q-3, R-4, S-1
(B) P-4, Q-2, R-3, S-1
(C) P-2, Q-4, R-3, S-1
(D) P-2, Q-4, R-1, S-3

**39.** Which sequence shows the correct path of a C program from typing to running?

(A) Editor, compiler, linker, loader
(B) Editor, linker, compiler, loader
(C) Compiler, editor, loader, linker
(D) Editor, loader, compiler, linker

**40.** A memory has $2^{16}$ bytes. Its size is

(A) 32 KB (B) 64 KB
(C) 128 KB (D) 16 KB

## Answer Key

| | | | | | |
|---|---|---|---|---|---|
| 1. **B** | 2. **C** | 3. **A** | 4. **C** | 5. **D** | 6. **B** |
| 7. **C** | 8. **A** | 9. **C** | 10. **D** | 11. **A** | 12. **D** |
| 13. **B** | 14. **A** | 15. **D** | 16. **C** | 17. **C** | 18. **A** |
| 19. **A** | 20. **B** | 21. **C** | 22. **B** | 23. **C** | 24. **D** |
| 25. **A** | 26. **A** | 27. **D** | 28. **B** | 29. **C** | 30. **B** |
| 31. **D** | 32. **C** | 33. **A** | 34. **B** | 35. **D** | 36. **A** |
| 37. **D** | 38. **C** | 39. **A** | 40. **B** | | |

## Explanations

**1. (B)** Babbage designed the Analytical Engine in the 1830s, which is why he is called the father of the computer. Pascal built an adding machine, the Pascaline.

**2. (C)** First-generation machines such as ENIAC used vacuum tubes. Transistors mark the second generation, ICs the third and microprocessors the fourth.

**3. (A)** Four bits make a nibble, and two nibbles make a byte of 8 bits.

**4. (C)** A scanner sends an image of a page into the computer. The other three give output.

**5. (D)** An interpreter works statement by statement. A compiler translates the whole program first, and an assembler handles assembly language only.

**6. (B)** The ALU does all arithmetic and logical operations. The control unit only directs the other units.

**7. (C)** The operating system runs the computer itself. The other three are application software, used to do the user's own work.

**8. (A)** The Centre for Development of Advanced Computing (C-DAC) built PARAM 8000 in 1991.

**9. (C)** Cheque numbers are printed in magnetic ink and read by a magnetic ink character reader. OMR reads marks, such as the bubbles on an answer sheet.

**10. (D)** The order is KB, MB, GB, TB, PB. Each unit is 1024 times the one before it.

**11. (A)** Firmware is held in ROM and stays when the power is off. The start-up program of a PC is an example.

**12. (D)** A dot matrix printer strikes pins against an inked ribbon. The other three print without striking the paper.

**13. (B)** Statement 2 is wrong. Integrated circuits mark the third generation; the fourth is marked by the microprocessor, a whole CPU on one chip.

**14. (A)** A compiler translates once and keeps the object code. An interpreter keeps no object code, so the translation is repeated on every run, and that costs time.

**15. (D)** OMR reads marked bubbles and MICR reads magnetic-ink numbers on cheques. A barcode reader reads product labels, and a plotter draws large plans and maps.

**16. (C)** The order is byte, KB, MB, GB, TB. Each step is a factor of 1024.

**17. (C)** Arithmetic is the work of the ALU. The control unit fetches, decodes and directs, but it does not calculate.

**18. (A)** ENIAC was unveiled in 1946 and the transistor came in 1947. The Intel 4004 followed in 1971 and the IBM PC in 1981.

**19. (A)** Statement 3 describes an interpreter. A compiler lists all the errors together after scanning the whole program, before anything runs.

**20. (B)** The linker builds the executable file. The loader comes after it and places that file in main memory.

**21. (C)** A touch screen shows output and also takes input by touch. A joystick and a scanner are input devices, and a plotter is an output device.

**22. (B)** 1 TB = 1024 GB and 1 GB = 1024 MB, so 1 TB = $2^{10} \times 2^{10} = 2^{20}$ MB. The figure $2^{40}$ is the number of bytes in a terabyte.

**23. (C)** Language translators are system software. R is false: preparing letters and accounts is done with application software.

**24. (D)** A spreadsheet is general-purpose application software. Drivers, operating systems and translators all serve the computer itself.

**25.** **(A)** The stored-program idea is that one memory holds both program and data. Statement 3 contradicts statement 1.

**26.** **(A)** Analog computers measure; digital computers count. Mainframe and micro are classes by size, not by the kind of data.

**27.** **(D)** A laser printer fixes powdered toner on the paper without striking it. The other three are impact printers.

**28.** **(B)** FLOPS means floating-point operations per second. dpi measures print quality and bps measures the speed of data transfer.

**29.** **(C)** In SQL the user states what result is wanted, not the steps to get it. Machine language is 1GL, assembly is 2GL and C is 3GL.

**30.** **(B)** Assembly language replaces binary codes with short words. Machine language has only 0s and 1s.

**31.** **(D)** A utility helps to maintain the system. A payroll package is application software; the compiler and the operating system are system software.

**32.** **(C)** A mainframe has very large storage and supports thousands of users at once. A microcomputer serves one user at a time.

**33.** **(A)** 32 GB = 32 × 1024 MB = 32,768 MB, and 32,768 ÷ 8 = 4096. Option (B) wrongly takes 1 GB as 1000 MB.

**34.** **(B)** 4 KB = 4 × 1024 = 4096 bytes, and 4096 × 8 = 32,768 bits. Option (A) is the number of bytes, not bits.

**35.** **(D)** A compiler makes no object code if there is an error, so nothing runs. An interpreter has already run every statement before the faulty one.

**36.** **(A)** Statement 1 is wrong: ROM keeps its contents without power. RAM is the volatile one.

**37.** **(D)** 4 GB = 4096 MB. Five CDs hold only 3500 MB, so a sixth is needed: 6 × 700 = 4200 MB.

**38.** **(C)** The order of components is vacuum tube, transistor, integrated circuit, microprocessor.

**39.** **(A)** The source code is typed in an editor and translated by the compiler. The linker then builds the executable file, and the loader puts it into memory.

**40.** **(B)** $2^{16} \div 2^{10} = 2^{6} = 64$, so the memory is 64 KB.