

11th Computer Science Text Book

Exercise Long Questions' Answers

According to the Full Book (Punjab Board, New Syllabus)

COMPUTER SCIENCE

AND

ENTREPRENEURSHIP

11

By: Atif Masih

Chapter 1: Introduction to Software Development

Long Question 1: Design a flowchart for a user registration process in a software application. Outline its key steps.

A flowchart is a diagram that shows the steps of a process in order, using standard symbols: oval = start/end, rectangle = process, parallelogram = input/output, diamond = decision (Yes/No), arrows = flow of control. The textbook shows the same notation for activity diagrams in Figure 1.8.

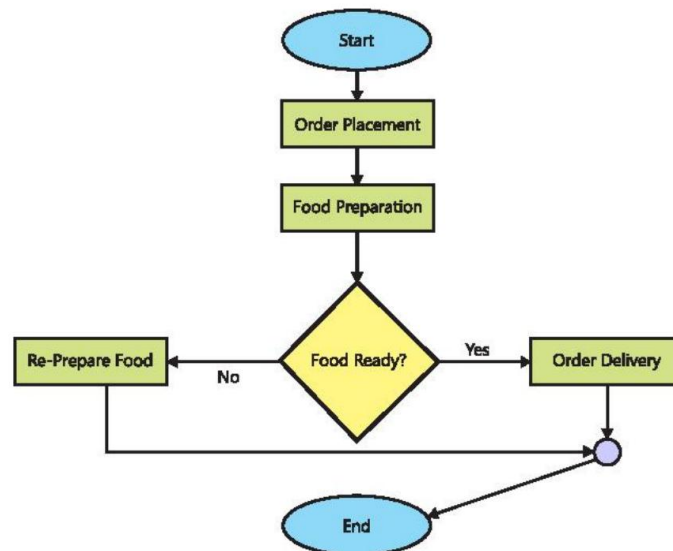
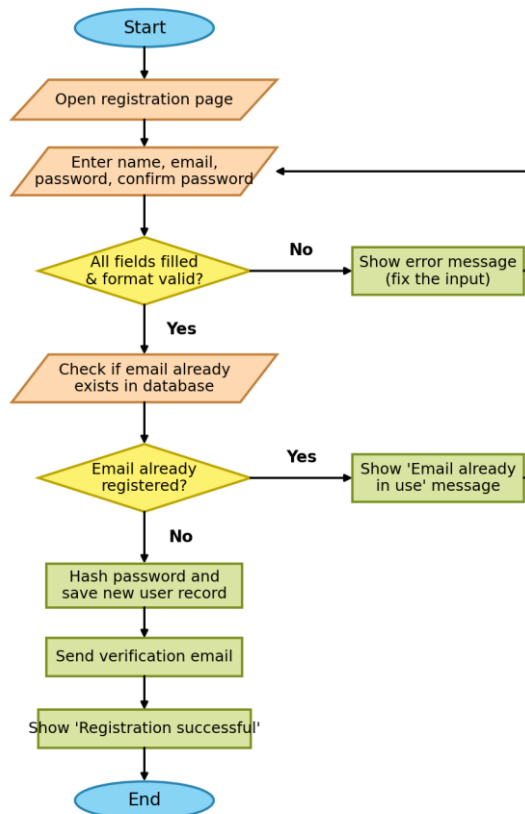


Figure 1.8: Activity Diagram with Decision and Connector Symbol

Flowchart of the user registration process

Flowchart: User Registration Process



Key steps

1. **Start** – the user opens the application and chooses Register / Sign up.
2. **Input** – the user enters name, email, password and confirm password.
3. **Validate input (decision)** – the system checks that all fields are filled and the email/password format is correct. If not, an error message is shown and the user re-enters the data.
4. **Check for duplicates (decision)** – the system searches the database; if the email is already registered, it shows an "already in use" message.
5. **Save the record** – the password is hashed and the new user record is stored.
6. **Verify** – a verification email is sent to confirm the address.
7. **Confirm and End** – a "Registration successful" message is displayed and the process ends.

Long Question 2: Imagine you are managing a project to develop a simple mobile application. Describe how you would use the Agile Methodology to handle this project.

Agile Methodology is a flexible and adaptive approach that delivers small, working parts of the software quickly and adapts to change. Work is done in short cycles called iterations or sprints, so

teams deliver parts of the software rapidly and gather feedback early (Figure 1.3). To develop a simple mobile application using Agile, I would follow these steps:



Figure 1.3: Agile Methodology

1. **Form a cross-functional team.** Include developers, a UI/UX designer, a product owner and a scrum master. The team is self-organizing and empowered to make decisions.
2. **Define the product backlog.** The product owner works with stakeholders to list features (user stories) by business value, e.g. "As a user, I want to sign up using my email", "...browse products by category", "...add items to a cart".
3. **Plan sprints (iterations).** Each sprint is 1–4 weeks. In sprint planning the team picks user stories from the top of the backlog that it can finish in the sprint.
4. **Hold daily stand-up meetings.** A 15-minute meeting: What did I do yesterday? What will I do today? Are there any impediments?
5. **Develop and test incrementally.** The team designs, codes and tests the chosen features. Agile practices from the textbook: continuous integration (merge code regularly to catch problems early), test-driven development (write tests before code) and pair programming (two developers at one workstation).
6. **Sprint review.** At the end of the sprint the team demonstrates the working increment to stakeholders and collects feedback into the backlog.
7. **Sprint retrospective.** The team reflects on what went well and what to improve in the next sprint.
8. **Repeat** until the product meets the desired quality and user needs.

Benefits and limitations (textbook)

Benefits	Limitations
High flexibility – requirements can change even after development has started	Scaling challenges – large projects with many teams need careful coordination
Improved customer satisfaction – regular delivery of working software and feedback	Stakeholder involvement – needs active participation from all stakeholders
Early delivery – a basic version (MVP) can be released quickly	Less predictable – timeline and final scope are harder to predict

Long Question 3: Consider an online banking system. Create a Use Case Diagram to show the interactions between customers, bank staff, and the system.

A Use Case Diagram shows what a system does from the user's point of view: which actors (people or other systems) interact with it and which use cases (goals) they can achieve. The textbook's example is a library system where Librarian and Student are actors and "Borrow book" and "Return book" are use cases (Figure 1.5).

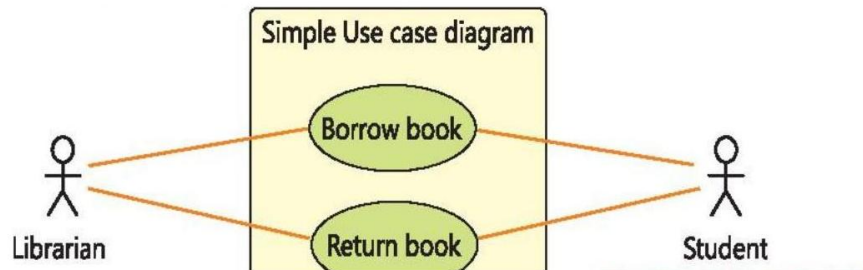
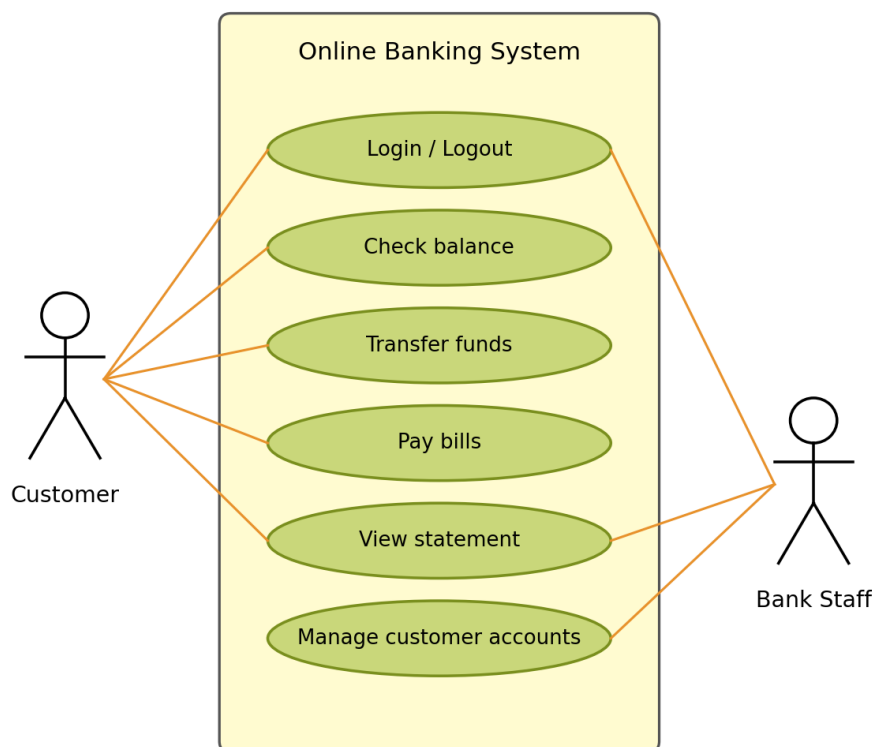


Figure 1.5: Example Use Case Diagram for a Library System

Following the textbook's steps for identifying use cases

- **Identify actors:** Customer and Bank Staff.
- **Define goals:** Customer – check balance, transfer funds, pay bills, view statement. Bank Staff – manage customer accounts and support customers.
- **Outline interactions:** both actors must log in first, so "Login / Logout" is shared by both.
- **Validate:** review the use cases with stakeholders (bank managers and customers).



Use Case Diagram: Online Banking System

Actor	Use cases
Customer	Login / Logout, Check balance, Transfer funds, Pay bills, View statement
Bank Staff	Login / Logout, Manage customer accounts, View statement

Long Question 4: You are developing a food delivery application. Create a Sequence Diagram to show the process of placing an order, from the customer selecting items to the delivery of the order.

A Sequence Diagram shows how objects interact in a particular order over time. Each participant has a vertical dashed lifeline and messages are horizontal arrows read from top to bottom. The textbook's example (Figure 1.7) shows a user opening boxes and putting items inside them.

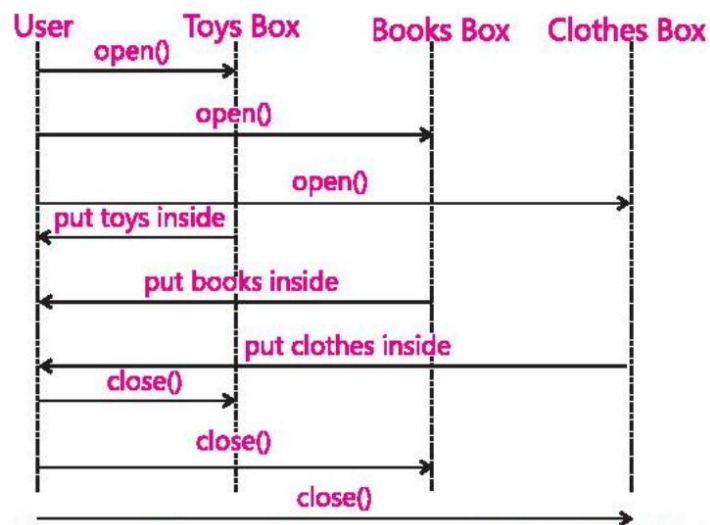
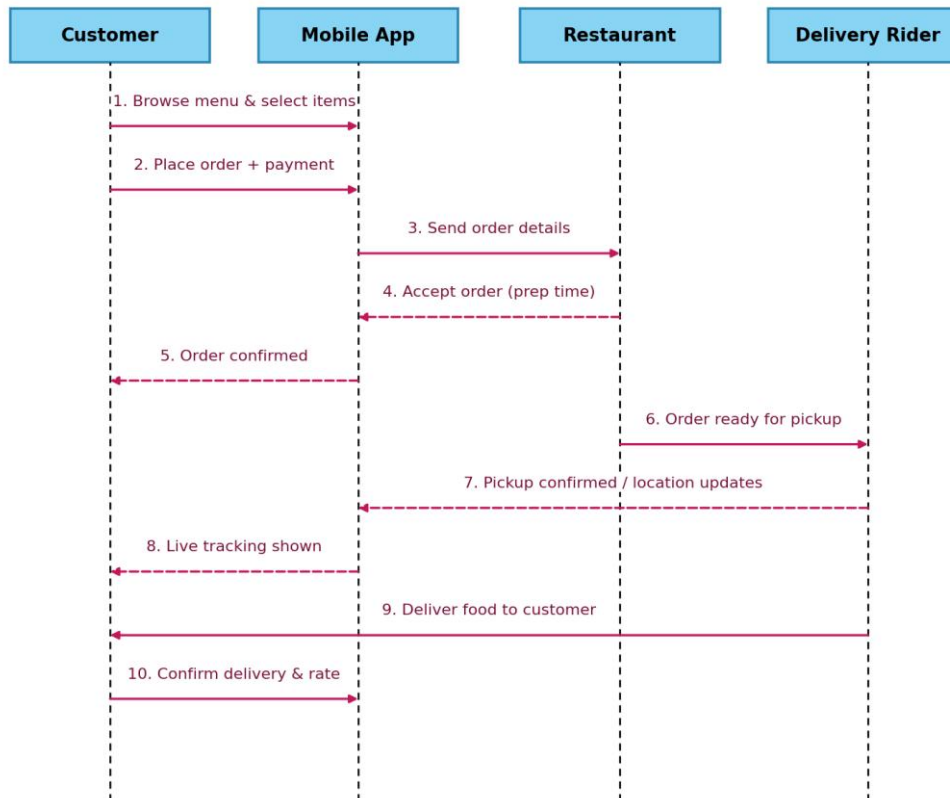


Figure 1.7: Sequence diagram of the user organizing items into labeled boxes

Sequence Diagram: Placing a Food Delivery Order



Sequence Diagram: Placing a Food Delivery Order

Explanation of the messages

1. The customer browses the menu and selects items in the mobile app.
2. The customer places the order and pays.
3. The app sends the order details to the restaurant.
4. The restaurant accepts the order and gives a preparation time.
5. The app confirms the order to the customer.
6. When the food is ready the restaurant notifies the delivery rider.
7. The rider confirms pickup and shares location updates.
8. The app shows live tracking to the customer.
9. The rider delivers the food to the customer.
10. The customer confirms delivery and rates the order.

Long Question 5: Discuss the importance of software development tools in the software development process. (a) Explain the role of language editors, translators, and debuggers in creating and maintaining software. (b) Provide examples of each tool and describe how they contribute to the efficiency and accuracy of software development.

Software development tools are applications that help developers create, test, debug and maintain software. They reduce manual effort, minimize errors and increase productivity.

a) Roles of key tools

- **Language editors** – provide an environment to write and edit source code. Examples from the textbook: Notepad++ (a simple but powerful code editor) and VS Code (popular, with many extensions). Modern editors offer syntax highlighting, auto-completion and error detection.
- **Translators** – convert code written in a high-level language into machine code (binary) that the computer can execute. Two types: interpreters translate line by line (e.g. Python) and compilers translate the whole program at once (e.g. GCC for C/C++).
- **Debuggers** – find and fix errors (bugs) in code by letting developers run it step by step, set breakpoints and inspect variables. Examples: GDB (GNU Debugger) for C/C++ and the Visual Studio Debugger.

b) Examples and contribution

Tool type	Example(s)	Contribution to efficiency & accuracy
Editor / IDE	Notepad++, VS Code, Visual Studio, PyCharm	Real-time syntax checking, code suggestions, project navigation. An IDE combines editor, compiler, debugger and version control in one place.
Compiler	GCC, Javac	Converts source code to machine code and reports syntax errors.
Interpreter	Python	Immediate execution; ideal for testing small pieces of code.
Debugger	GDB, Visual Studio Debugger	Step-through execution, variable inspection, finds the exact cause of logical errors.
Source code repository	GitHub, Bitbucket	Stores and tracks code changes so many developers can work together without conflicts.

Chapter 2: Python Programming

Long Question 1: Evaluate the following Python expressions. (a) $(18 / 3 + 4 ** 2) - (2 * (7 - 3)) / (9 - 7.4)$ (b) $(25 + 3 * 4 ** 2 - 6) / (2 ** 3 + 1) - 7$ (c) $(12 + 6 * (5 - 2)) ** 2 / ((4 ** 2 - 7) + 10)$ (d) $45 / (2 ** 2 + 3 * 4) + 8 * (7 - 3)$

Python follows the order of operations: parentheses first, then exponent (**), then multiplication and division (left to right), then addition and subtraction.

Note: in part (a) the textbook prints the last bracket as "(9 - 7,4)". It is read here as (9 - 7.4), with the comma as a decimal point.

(a) $(18 / 3 + 4 ** 2) - (2 * (7 - 3)) / (9 - 7.4)$

- $18 / 3 = 6.0$
- $4 ** 2 = 16$
- $6.0 + 16 = 22.0$
- $7 - 3 = 4$, then $2 * 4 = 8$
- $9 - 7.4 = 1.6$
- $8 / 1.6 = 5.0$
- $22.0 - 5.0 = 17.0$

Result: 17.0

(b) $(25 + 3 * 4 ** 2 - 6) / (2 ** 3 + 1) - 7$

- $4 ** 2 = 16$
- $3 * 16 = 48$
- $25 + 48 - 6 = 67$
- $2 ** 3 = 8$, then $8 + 1 = 9$
- $67 / 9 = 7.444444444444445$
- $7.444444444444445 - 7 = 0.44444444444444464$

Result: 0.44444444444444464

(c) $(12 + 6 * (5 - 2)) ** 2 / ((4 ** 2 - 7) + 10)$

- $5 - 2 = 3$
- $6 * 3 = 18$
- $12 + 18 = 30$
- $30 ** 2 = 900$
- $4 ** 2 = 16$
- $16 - 7 = 9$
- $9 + 10 = 19$
- $900 / 19 = 47.368421052631575$

Result: ≈ 47.3684 (Python prints 47.36842105263158)

(d) $45 / (2 ** 2 + 3 * 4) + 8 * (7 - 3)$

- $2 ** 2 = 4$
- $3 * 4 = 12$
- $4 + 12 = 16$
- $45 / 16 = 2.8125$
- $7 - 3 = 4$
- $8 * 4 = 32$
- $2.8125 + 32 = 34.8125$

Result: 34.8125

Checking in Python

```
print((18 / 3 + 4 ** 2) - (2 * (7 - 3)) / (9 - 7.4)) # 17.0
print((25 + 3 * 4 ** 2 - 6) / (2 ** 3 + 1) - 7) # 0.44444444444444464
print((12 + 6 * (5 - 2)) ** 2 / ((4 ** 2 - 7) + 10)) # 47.36842105263158
print(45 / (2 ** 2 + 3 * 4) + 8 * (7 - 3)) # 34.8125
```

Long Question 2: Translate the following mathematical expressions to Python syntax.

(a) $5 \times (3 + 2^2)$ (b) $6 - 2 \times 3$ (c) $7 + 2^2$

Mathematical expression	Python syntax	Result
(a) $5 \times (3 + 2^2)$	$5 * (3 + 2 ** 2)$	35
(b) $6 - 2 \times 3$	$6 - 2 * 3$	0
(c) $7 + 2^2$	$7 + 2 ** 2$	11

General rules: multiplication $\times$ becomes $*$, division $\div$ becomes $/$ (or $//$ for floor division), and exponent (power) becomes $**$. Use parentheses to keep the correct order of operations.

Long Question 3: Explain the concept of variables in Python.

A variable is a storage container in the computer's memory that holds data such as numbers or text. The value stored in a variable can change while the program runs.

- **Dynamic typing** – no need to declare the data type; it is decided from the value assigned.
- **Case-sensitive** – age and Age are different variables.
- **Naming rules (textbook)** – the name must begin with a letter (A–Z, a–z) or an underscore (`_`); it may then contain letters, digits (0–9) or underscores; names are case-sensitive; Python keywords such as `if`, `for`, `while` cannot be used as names.
- **Rebinding** – the same variable can later hold a value of another type.
- **Types of variables** – integer (`int`), floating-point (`float`), string (`str`) and Boolean (`bool`).

```
age = 17 # integer
price = 19.99 # float
name = "Ali" # string
is_student = True # boolean
```

Tip from the textbook: use meaningful variable names so the code is easy to understand.

Long Question 4: Write a Python program that takes a number as input and checks whether it is positive, negative, or zero using an if-elif-else statement.

```
# Program to check if a number is positive, negative, or zero
num = float(input("Enter a number: "))

if num > 0:
    print("The number is positive.")
elif num < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

Explanation: input() reads the value as text, and float() converts it so decimals also work. The if-elif-else statement tests the conditions in order and runs only the first block that is true.

Long Question 5: Write a Python program using a while loop that prints all the odd numbers between 1 and 100. Also, count and print the total number of odd numbers.

```
# Print odd numbers from 1 to 100 and count them
number = 1
count = 0

while number <= 100:
    if number % 2 != 0:
        print(number)
        count += 1
    number += 1

print("Total odd numbers:", count)
```

Output: the odd numbers 1, 3, 5, ... 99 are printed, followed by Total odd numbers: 50

Explanation: the loop runs while number <= 100. The condition number % 2 != 0 is true when the remainder is not zero, i.e. the number is odd. The counter increases each time and the total is printed at the end.

Chapter 3: Algorithms and Problem Solving

Long Question 1: Provide a detailed explanation of why the Halting Problem is considered unsolvable and its implications in computer science.

Textbook definition: a problem is solvable if an algorithm can solve it within a finite amount of time; it is unsolvable if no algorithm can be created that gives a solution in all cases. The Halting Problem is the textbook's example of an unsolvable problem.

What is the Halting Problem?

It asks: given any program and its input, can we determine whether the program will eventually halt (finish running) or continue to run forever? Alan Turing proved that no general algorithm can solve the Halting Problem for all possible program-input pairs.

Why it is unsolvable (idea of Turing's proof)

1. Assume a perfect program $\text{HALT}(P, I)$ exists that always answers "halts" or "loops forever" for any program P and input I .
2. Build a new program $\text{TROUBLE}(P)$ that calls $\text{HALT}(P, P)$. If HALT says "halts", TROUBLE loops forever; if HALT says "loops forever", TROUBLE stops.
3. Now run TROUBLE on itself. If HALT says $\text{TROUBLE}(\text{TROUBLE})$ halts, then TROUBLE loops forever. If HALT says it loops, then TROUBLE halts.
4. Either way HALT gives the wrong answer – a contradiction. So a perfect HALT program cannot exist.

Implications in computer science

- Some problems can never be solved by any algorithm, however powerful the computer or clever the programmer – computation has fundamental limits.
- It is impossible to write a perfect tool that detects all infinite loops or proves every program correct.
- Related questions about program behavior (for example, whether two programs do the same thing) are also undecidable.
- It classifies problems as solvable or unsolvable, so programmers do not waste time on impossible tasks (textbook tidbit: first decide whether the problem is solvable).
- Practical tools work only for restricted cases or use approximations and heuristics.

Long Question 2: Discuss the characteristics of search problems and compare the efficiency of Linear Search and Binary Search algorithm.

Search problems involve finding a particular element (or set of elements) in a collection of data. They are fundamental in databases, information retrieval and AI.

Characteristics of search problems

- **Input:** a collection (array, list, tree, graph) and a target value.
- **Output:** the position (index) of the target, or a confirmation that it is present or absent.
- **Evaluation:** measured by the number of comparisons or steps.

Linear Search (textbook)

Checks each item one by one until the target is found or the list ends: start at the beginning, check each item, move to the next, repeat. Textbook example: in the list [Karachi, Lahore, Islamabad, Faisalabad], searching for Islamabad checks Karachi, then Lahore, then finds Islamabad.

- Time: best $O(1)$, worst $O(n)$, average $O(n)$; space $O(1)$.
- Works on unsorted data; best for small datasets or data that changes often.

Binary Search (textbook)

Works on a sorted list by repeatedly dividing the search interval in half: compare the target with the middle element, and discard the half that cannot contain it.

- Time: best $O(1)$, worst/average $O(\log n)$; space $O(1)$ (iterative).
- Requires sorted data; best for large, static datasets.

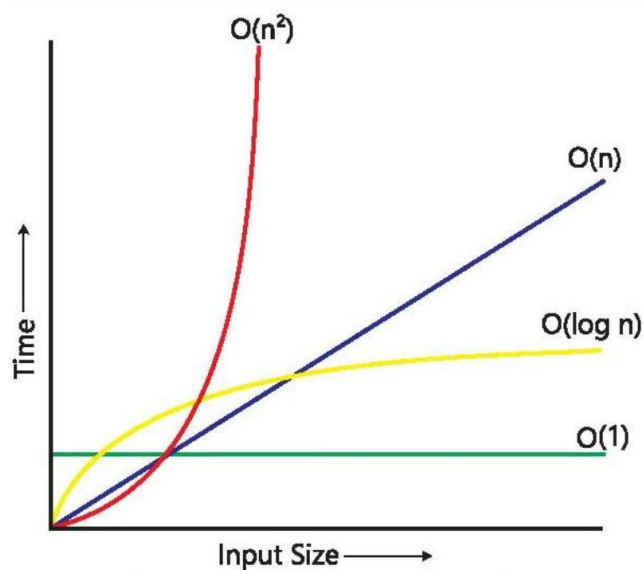


Figure 3.5: Growth of asymptotic notations – $O(n)$ grows steadily while $O(\log n)$ grows very slowly

Criterion	Linear Search	Binary Search
Data requirement	Unsorted or sorted	Sorted only
Time complexity	$O(n)$	$O(\log n)$
Efficiency on large data	Poor	Excellent
Implementation	Simple	Slightly complex

Conclusion: Binary Search is far more efficient on large sorted datasets, but Linear Search is more flexible because it needs no preprocessing.

Long Question 3: Discuss the nature of optimization problems and provide examples of their applications in real-world scenarios.

Optimization problems ask for the best solution among all feasible solutions according to some criterion (minimize cost, maximize profit, shortest path). They involve an objective function

(measures quality), constraints (conditions that must be satisfied) and decision variables (values that can be changed).

- **Continuous optimization** – variables take any real value (e.g. linear programming).
- **Discrete (combinatorial) optimization** – variables are discrete (e.g. scheduling, routing).

Real-world applications

- **Traveling Salesman Problem (TSP)** – find the shortest route visiting a set of cities and returning to the start (logistics, circuit-board drilling). The textbook notes TSP is NP-hard.
- **Knapsack Problem** – choose items of maximum total value without exceeding the weight capacity (cargo loading, resource allocation).
- **Job scheduling** – assign jobs to machines to minimize completion time (manufacturing, cloud computing).
- **Network flow** – maximize data flow through a network (telecommunications, traffic engineering).
- **Supply chain management and portfolio optimization** – reduce cost and inventory; maximize return for a given risk.

Solution approaches (textbook algorithm design techniques)

- **Greedy algorithms** – make the locally best choice at each step (e.g. the Coin Change problem: choose the largest coin that does not exceed the remaining amount).
- **Dynamic programming** – break the problem into overlapping subproblems and store results to avoid repeated work (e.g. Fibonacci numbers).
- **Exact methods** (branch and bound) and **approximate methods** (genetic algorithms, simulated annealing) are also used.

Long Question 4: Explain the process and time complexity of the Bubble Sort algorithm. Compare it with another sorting algorithm of your choice in terms of efficiency.

Bubble Sort is one of the simplest sorting algorithms. It repeatedly steps through the list, compares adjacent elements and swaps them if they are in the wrong order.

Process (textbook)

- Start from the beginning of the list.
- Compare each pair of adjacent elements.
- Swap them if they are in the wrong order.
- Continue passing through the list until no swaps are needed.

Textbook example: for the list [5, 3, 8, 4, 2], Bubble Sort first compares 5 and 3 and swaps them, then moves to the next pair (5 and 8), and so on. After several passes the list is sorted as [2, 3, 4, 5, 8].

Time complexity: worst and average $O(n^2)$; best $O(n)$ if the list is already sorted (with a no-swap flag). **Space complexity:** $O(1)$ – it sorts in place.

Comparison with Merge Sort

Merge Sort uses divide and conquer: it splits the list into halves, sorts each half and merges them. The textbook's Figure 3.6 shows the dividing steps for the list [14, 7, 3, 12].

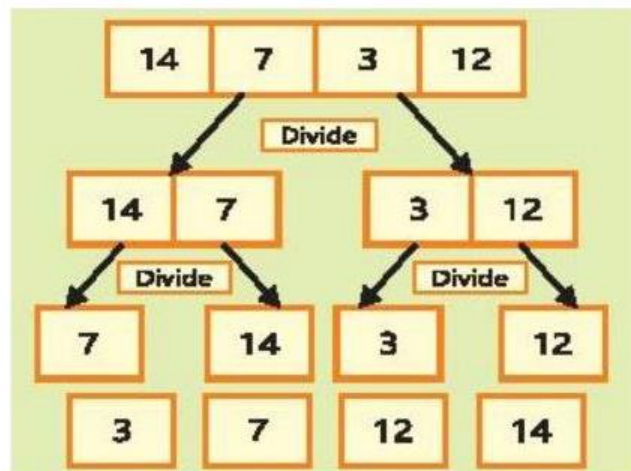


Figure 3.6: Merge Sort process

Feature	Bubble Sort	Merge Sort
Approach	Compare and swap adjacent elements	Divide and conquer
Time (worst)	$O(n^2)$	$O(n \log n)$
Time (best)	$O(n)$	$O(n \log n)$
Space	$O(1)$	$O(n)$ (auxiliary array)
Stable?	Yes	Yes
Suitable for large data	No	Yes

Conclusion: Merge Sort runs in $O(n \log n)$ time and suits large datasets, while Bubble Sort's quadratic time becomes impractical as n grows. Bubble Sort is mainly educational.

Long Question 5: Discuss the differences between time complexity and space complexity. How do they impact the choice of an algorithm for a specific problem?

Time complexity measures how the running time of an algorithm grows as the input size (n) increases, written in Big O notation. Space complexity measures how the amount of memory used grows as the input size increases.

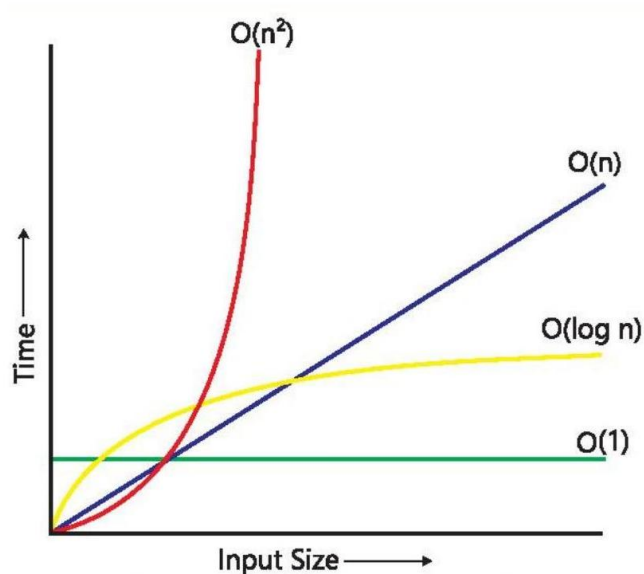


Figure 3.5: Growth of asymptotic notations – $O(1)$, $O(\log n)$, $O(n)$, $O(n^2)$

Aspect	Time complexity	Space complexity
What it measures	Execution time (number of operations)	Memory used
Impact	Responsiveness and speed	Hardware requirements and scalability
Trade-off	Can often trade space for time	Can often trade time for space

Big O notations in the textbook

- $O(1)$ – constant time; $O(\log n)$ – logarithmic (binary search); $O(n)$ – linear (linear search); $O(n^2)$ – quadratic (comparing every pair of items).

Impact on choosing an algorithm

- **Limited memory (embedded systems):** prefer low space complexity even if slower.
- **Real-time systems:** prefer the fastest algorithm even if it uses more memory.
- **Big data:** prefer $O(n \log n)$ over $O(n^2)$, even if extra space is needed.
- **Trade-offs:** dynamic programming uses extra memory to greatly reduce running time.

Examples: Merge Sort – $O(n \log n)$ time, $O(n)$ space; Bubble Sort – $O(n^2)$ time, $O(1)$ space. The best choice balances both according to the problem's constraints.

Chapter 4: Computational Structures

Long Question 1: Discuss the dynamic size property of lists in Python. How does this property make lists more flexible?

A Python list can grow or shrink automatically while the program runs. Unlike fixed-size arrays in languages such as C or Java, a list does not need a predefined size. Textbook properties of lists: (1) Dynamic size – you can add items to the list or remove items without any problem; the list adjusts its size automatically. (2) Index-based access – every item has a position (index) starting from 0. (3) Ordered collection – items stay in the position where you placed them.

How it works

Internally a list is a dynamic array. When an element is appended and the underlying array is full, Python allocates a larger array and copies the existing elements. Removing elements can shrink the list.

Textbook examples of changing a list

Textbook example – removing by index (the list shrinks)

```
party_list = ["Invite friends ", "Buy decorations", "Buy cold drinks "]
party_list.pop(0)      # Removes the item at index 0
print(party_list)
# Output: ['Buy decorations', 'Buy cold drinks']
```

Textbook example – searching with the 'in' keyword

```
party_list = ["Invite friends", "Buy decorations", "Buy cold drinks"]
if "Buy cold drinks" in party_list:
    print("Buy cold drinks is on the list.")
else:
    print("Buy cold drinks is not on the list.")
# Output: Buy cold drinks is on the list.
```

Simple illustration of growth using append()

```
numbers = []           # empty list
numbers.append(10)      # size 1
numbers.append(20)      # size 2
numbers.append(30)      # size 3
print(numbers)          # [10, 20, 30]
```

Why this makes lists flexible

- **No pre-allocation** – the programmer does not have to guess the maximum size.
- **Easy insertion and deletion** with `append()`, `insert()`, `remove()` and `pop()`.
- A list can hold items of different data types.
- Works smoothly with loops and built-in functions.
- Resizing cost is spread out, so `append()` is $O(1)$ on average.

Long Question 2: Explain the operations on stack with real-life example and Python code.

A stack is a simple data structure where you can only add or remove items from one end, known as the "top". It works on the Last-In, First-Out (LIFO) principle: the most recently added element is the first one removed.



Figure 4.1: Stack of Books

Stack operations (textbook)

- **Push operation** – adding an item to the top of the stack.
- **Pop operation** – removing the item from the top of the stack.
- **Peek/Top (optional)** – viewing the top item without removing it.

Real-life example: a stack of books or plates – the last one placed on top is the first one taken off.

Textbook example – stack using a list

```
# Create an empty stack of books
stack_of_books = []
print("Initial stack:", stack_of_books)           # Empty stack

# Add books to the stack (push operation)
print("\nAdding books to the stack (push operation):")
stack_of_books.append('Book A')
print("Stack after pushing 'Book A':", stack_of_books)
stack_of_books.append('Book B')
print("Stack after pushing 'Book B':", stack_of_books)

# Remove the top book from the stack (pop operation)
print("\nDeletion of top book (pop operation):")
top_book = stack_of_books.pop()
print("Removed book:", top_book)
print("Stack after popping the top book:", stack_of_books)
```

Explanation (textbook): the code creates an empty stack, adds "Book A" and "Book B" one by one to the top, then removes the top book "Book B", showing that the last book added is the first one taken off.

Applications of stacks

- Function call management (call stack)
- Undo mechanisms in editors
- Expression evaluation and parenthesis matching
- Backtracking algorithms (maze solving, DFS)

Long Question 3: Write a simple program to implement a queue (insertion and deletion).

A queue is like a line in front of a bank or a ticket counter. The first person to get in line is the first to be served. It follows the FIFO (First-In, First-Out) principle: items are added at the back and removed from the front (Figure 4.2).

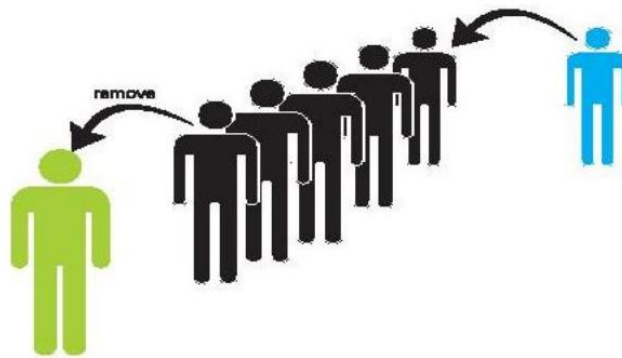


Figure 4.2: Queue of persons in front of the bank

Queue operations (textbook)

- **Enqueue (add an item)** – adds an item to the back (end) of the queue.
- **Dequeue (remove an item)** – removes the item at the front of the queue.
- Other operations: check if the queue is empty, view the front element without removing it (peek), and find the size of the queue.

Textbook example – queue using the queue module

```
# Built-in module to implement queues in Python
from queue import Queue

# Create a new queue
q = Queue()

# Add people to the queue (Enqueue)
q.put("Ahmed")      # Adds Ahmed to the end of the queue
q.put("Fatima")     # Adds Fatima to the end of the queue

# View the person at the front of the queue (Peek)
front_person = q.queue[0]  # Looks at the person at the front without removing them
print(front_person)

# Remove a person from the front of the queue (Dequeue)
removed_person = q.get()    # Removes and returns the person at the front of the queue
print(removed_person)

# Add another person to the queue (Enqueue)
q.put("Sara")         # Adds Sara to the end of the queue

# View the updated queue
updated_queue = list(q.queue)
print(updated_queue)
```

Explanation (textbook): the code manages a line of people using a queue. It adds people to the end, checks who is at the front without removing them, serves (removes) the person at the front, then adds another person and shows the updated line.

Output: Ahmed, then Ahmed, then ['Fatima', 'Sara'].

Applications of queues

- CPU scheduling and print spooling
- Breadth-First Search (BFS)
- Handling requests in web servers

Long Question 4: Define Tree and explain its properties.

A tree data structure organizes information in a way that spreads out from a main point called the root node. Each piece of information is a node, and nodes connect to other nodes, forming a branching structure. This is different from a list, where items are organized one after the other in a straight line. Example: in a family tree the oldest ancestors represent the root node, and each individual may have descendants forming lower levels (Figure 4.3).

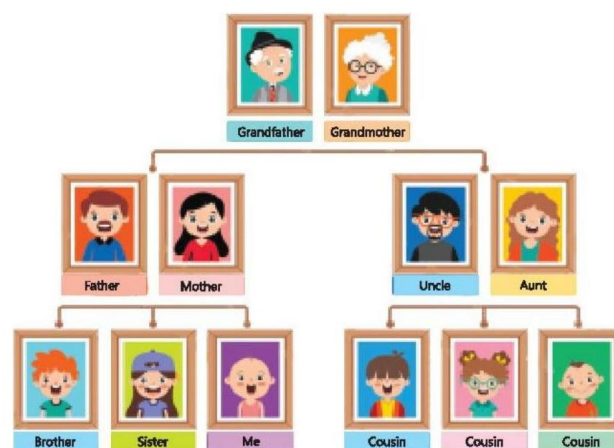


Figure 4.3: Family Tree

Properties of trees (textbook)

1. **Root node** – the very first or top node in a tree, like the main folder in a computer where all other folders and files are contained.
2. **Edges and nodes** – nodes are the individual elements; they are connected by lines called edges. A node without any child nodes is called a leaf, like a file in a folder that contains no other files.
3. **Height** – the longest path from the root node down to the farthest leaf; it tells how deep or tall the tree is.
4. **Balanced trees** – a tree is balanced if the branches on the left and right sides are nearly the same height.

Other properties

- There is exactly one path between any two nodes, and a tree has no cycles (loops).
- A tree with n nodes has exactly $n - 1$ edges.
- Types: Binary Tree (at most two children per node), Binary Search Tree, Balanced Tree, Heap.

Applications of trees (textbook)

- **File systems** – pre-order traversal is useful for backing up directories before their contents.
- **File system deletion** – post-order traversal deletes sub-directories and files before the parent directory.
- **Hierarchical data representation** – organizational charts and family trees.
- **Decision making** – decision trees in algorithms.

Long Question 5: What is a graph? Explain differences between directed and undirected graphs.

A graph is a data structure consisting of a set of vertices (or nodes) connected by edges. Graphs represent networks of connections, where each connection is a relationship between two vertices. Example: mapping the cities of Pakistan – each city is a vertex and each road between two cities is an edge. Formally $G = (V, E)$.

Difference from a tree (textbook)

- A tree is hierarchical with a single root; a graph does not necessarily have a hierarchy or a root.
- In a tree there is exactly one path between any two nodes (no cycles); in a graph there can be multiple paths and cycles.
- Trees represent structured data such as family trees; graphs can represent networks, web links or transport systems.

Properties of graphs (textbook)

- **Degree** – number of edges connected to a vertex.
- **Weight** – a value on an edge such as distance or cost.
- **Direction** – edges can be directed (one-way) or undirected (two-way).

Directed graph

Edges have a direction – they go from one vertex to another in a specific way (Figure 4.4). If there is no one-way street from city A to city B, you cannot travel directly from A to B.

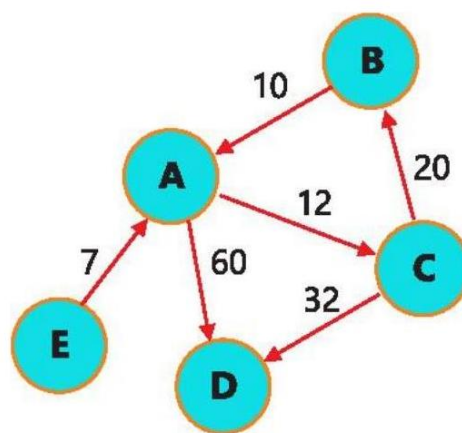


Figure 4.4: Directed Weighted Graph

Undirected graph

Edges have no direction – if there is a connection between two vertices you can travel in both directions (Figure 4.5). If Person A is friends with Person B, then B is also friends with A.

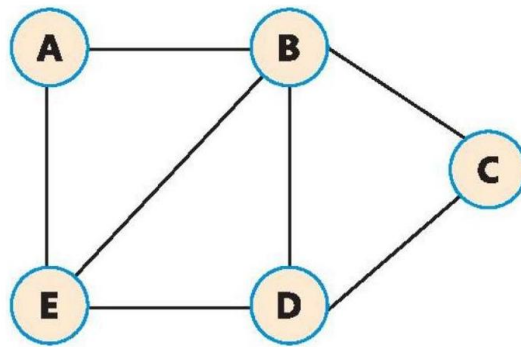


Figure 4.5: Undirected Graph

Feature	Directed graph	Undirected graph
Edge	One-way ($u \rightarrow v$)	Two-way ($u - v$)
Symmetry	Not symmetric	Symmetric
Example	Web page links, flight routes, one-way roads	Friendship in a social network, two-way roads
Degree	In-degree and out-degree	Degree only
Traversal	Must follow the edge direction	Can travel both ways

Weighted graphs (either type) attach a weight to each edge to represent distance, time or cost – useful for finding the shortest or quickest route.

Chapter 5: Data Analytics

Long Question 1: Explain the role and importance of statistical models in solving real-world problems.

Statistical modeling is used to analyze data to make sense of the real world and to predict what will happen in the future. Textbook example: to know how much money you will spend on groceries next month, you can look at your past expenses; by analyzing that data you can create a model to estimate your future expenses.

Role of statistical models

- **Understanding relationships** – models such as linear regression show how one variable (e.g. number of customers) affects another (daily earnings).
- **Prediction** – a model built from past data can forecast future values, e.g. earnings for tomorrow.
- **Classification** – logistic regression predicts a yes/no outcome, e.g. whether a student will pass or fail based on hours studied.
- **Grouping** – clustering (such as K-means) groups similar items, e.g. students by their performance in Math and English.
- **Drawing conclusions** – if hours studied strongly affect exam scores, students should study more to improve.

Importance

- Decision-making is based on evidence instead of guesswork.
- Reduces risk and cost through better planning (budgeting, stock, staffing).
- Used in many fields: business, healthcare, education, weather and sports.
- Model performance can be measured with error and accuracy metrics, so results can be trusted and improved.
- Ethical use: models must be fair and unbiased, and personal data must be kept private and secure.

Long Question 2: Describe the steps involved in building a basic statistical model (e.g., linear regression). Include details on data collection, model training, and evaluation.

General steps in model development (textbook)

1. **Define the problem** – understand what is to be predicted and which factors influence it (e.g. for grocery expenses: family size, location, income).
2. **Collect data** – gather data related to the problem, such as past spending habits.
3. **Choose an algorithm** – pick a suitable method such as linear regression or logistic regression.
4. **Train the model** – the model learns from the collected data to make predictions.
5. **Evaluate the model** – test it on new, unseen data to check that it makes accurate predictions.

Linear regression – textbook example

Linear regression helps understand the relationship between two variables and predict one variable from another. Example: a fruit stall owner predicts daily earnings from the number of customers. Number of customers is the independent variable (X) and daily earnings is the dependent variable (Y).

Step 1 – Collecting data (past 5 days):

Number of customers	Daily earnings (Rs.)
10	500
15	700
20	900
25	1,100
30	1,300

Step 2 – The formula: $Y = \beta_0 + \beta_1 X + \epsilon$

- Y is the dependent variable (daily earnings); X is the independent variable (number of customers).
- β_0 is the intercept – the value of Y when X = 0.
- β_1 is the slope – how much Y changes for each unit change in X.
- ϵ is the error term – the difference between predicted and actual values.

Step 3 – Building (training) the model:

- Slope: for every 5 extra customers earnings rise by 200 rupees, so $\beta_1 = 200 / 5 = 40$ (each customer adds Rs. 40).
- Intercept: using 10 customers and Rs. 500 $\rightarrow 500 = \beta_0 + (40 \times 10) \rightarrow \beta_0 = 500 - 400 = 100$ (base earnings even if no customer comes).
- Final equation: **Earnings = 100 + 40 × Customers**

Step 4 – Interpreting the model: with 22 expected customers, Earnings = $100 + (40 \times 22) = 100 + 880 = \text{Rs. } 980$.

Step 5 – Testing (evaluation): on day 6, 28 customers came and earned Rs. 1,250. The model predicts $100 + (40 \times 28) = \text{Rs. } 1,220$. The error = $1,220 - 1,250 = -30$ Rs. The prediction is close but not perfect, showing that real-world data has some variation.

Improving the model (textbook tidbit)

- Use more data points for better accuracy.
- Include relevant factors such as family size or special events.
- Regularly update the model with new data.
- Test predictions against actual values to refine the approach.

Long Question 3: Discuss the types of data visualizations and their uses.

Data visualization is the process of representing data in a visual format such as graphs or charts. It helps us quickly identify patterns, trends and insights from data. Different chart types serve different purposes.

1. Bar chart

Use: compare quantities across categories; the height (or length) of each bar shows the value of that category. Example: comparing the sales figures of different products in a store.

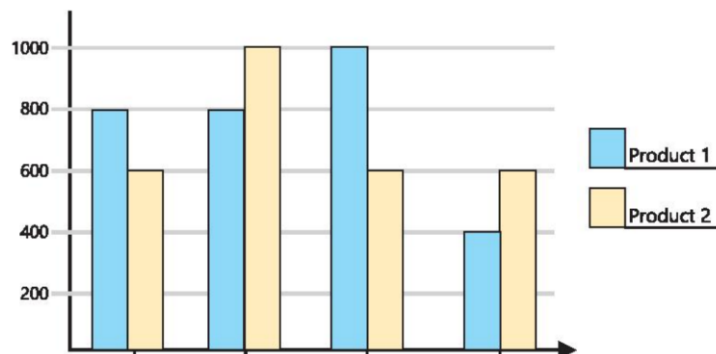


Figure 5.1: A bar chart showing sales figures of different products

2. Line graph

Use: show trends over time by plotting data points and joining them with a line. Example: temperature changing during the day or over a week.

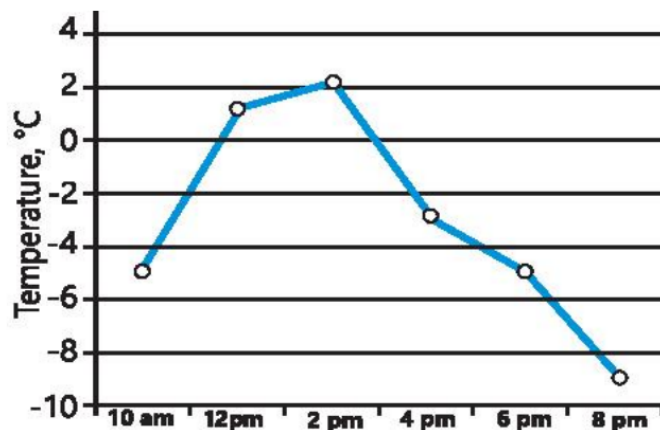


Figure 5.2: A line graph showing variation of temperature over time

3. Histogram

Use: show the distribution of a dataset by grouping data into bins or intervals, so you can see how often values occur in each range. Example: the distribution of students' scores in a math exam.

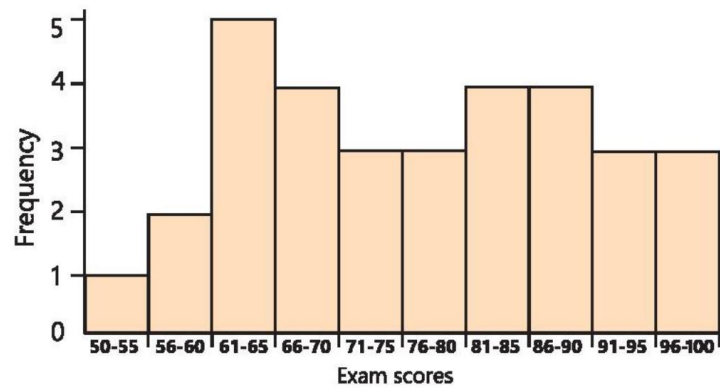


Figure 5.3: Example of a histogram showing the distribution of exam scores

4. Scatterplot

Use: display the relationship between two variables; each point is one observation. Example: the relationship between the number of hours studied and exam scores.

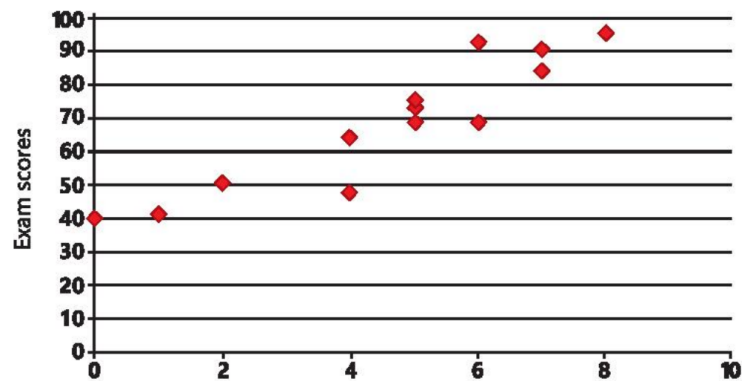


Figure 5.4: Scatterplot showing the relationship between hours studied and exam scores

5. Boxplot (whisker plot)

Use: summarize data distribution by displaying the median, quartiles and potential outliers. Example: comparing the exam scores of different classes to see which class performed better overall.

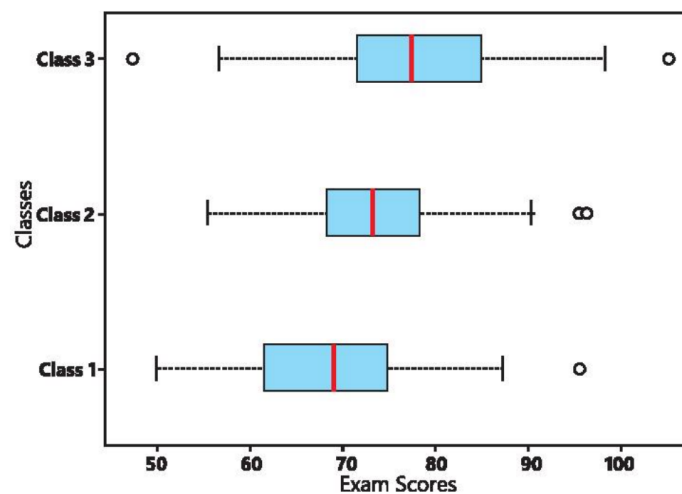


Figure 5.5: A boxplot showing class scores performance of three classes

Other common types

- **Pie chart** – shows proportions of a whole (e.g. market share).
- **Heatmap** – shows intensity of values with colours.
- **Area chart** – emphasizes the size of change over time.
- **Bubble chart** – a scatterplot where bubble size shows a third variable.

Choosing the right visualization

Purpose	Suitable chart
Comparison	Bar chart, boxplot
Trend over time	Line graph, area chart
Distribution	Histogram, boxplot
Relationship	Scatterplot, bubble chart
Composition	Pie chart, stacked bar chart

Long Question 4: Explain data collection methods.

Data collection is the process of gathering relevant information for a particular purpose in a structured way. The choice of method depends on the research question, resources and the nature of the data.

Primary data collection methods

- **Surveys / questionnaires** – a structured set of predefined questions given to a sample group (online, paper or by telephone). Efficient for large samples; limited depth.
- **Interviews** – one-on-one conversation (structured, semi-structured or unstructured). Gives rich qualitative data but is time-consuming and may have interviewer bias.
- **Observations** – watching people or events in their natural environment. Useful for behaviour studies; the observer may influence the subjects.
- **Experiments** – controlled manipulation of variables to find cause and effect. High internal validity but may lack realism.
- **Focus groups** – a small group discussion led by a moderator; gives diverse opinions but may not represent the larger population.

Secondary data collection

Using existing data from government reports, academic journals, company records and online databases. It is cost-effective and quick but may not fit the research need perfectly.

Factors influencing the choice

- Purpose of the study
- Budget and time
- Required accuracy
- Ethical considerations (privacy, consent)

Long Question 5: Discuss the concept of measure of tendency with example.

Measures of central tendency are single values that describe a set of data by identifying its central position. The three main measures are the mean, median and mode.

- **Mean (average)** – sum of all values divided by the number of values; sensitive to outliers.
Example: scores 50, 60, 70, 80, 90 → mean = $350 / 5 = 70$.
- **Median** – the middle value when data is arranged in order (average of the two middle values if the count is even); not affected by outliers. Example: [50, 60, 70, 80, 90] → median = 70; [50, 60, 70, 80] → median = $(60 + 70) / 2 = 65$.
- **Mode** – the value that appears most often; there can be one mode, more than one, or none.
Example: [50, 60, 70, 70, 90] → mode = 70; [50, 60, 70, 70, 60] → modes 60 and 70.

When to use each

- **Mean** – symmetric data without outliers.
- **Median** – skewed data or when outliers are present (e.g. household income).
- **Mode** – categorical data (e.g. the most common product sold).

Example scenario: a teacher wants the typical student performance. If one score is extremely high or low, the median gives a better picture than the mean.

Chapter 6: Emerging Technologies

Long Question 1: Define cloud deployment models and assess the differences among them.

Cloud deployment models define how cloud services are made available and used. Each model offers different levels of control, security and flexibility. The main models are Public, Private, Hybrid and Multi-cloud.

- **Public cloud** – offered over the internet and shared among multiple organizations, managed by a third-party provider. Example: Amazon Web Services (AWS). Pros: cost-effective, scalable. Cons: less control and less secure.
- **Private cloud** – used exclusively by one organization; hosted on-premises or by a third party but not shared. Example: a large bank handling sensitive customer data. Pros: high security and control. Cons: expensive.
- **Hybrid cloud** – combines public and private clouds and lets data and applications be shared between them. Example: a private cloud for sensitive data and a public cloud for everyday operations or busy periods. Pros: flexibility and control. Cons: complex to manage.
- **Multi-cloud** – uses services from multiple providers (AWS, Microsoft Azure, Google Cloud) at the same time. Example: a global retail company uses AWS for its e-commerce website, Azure for internal ERP applications and Google Cloud Platform for analytics and machine learning. Pros: avoids vendor lock-in and improves resilience. Cons: governance complexity.

Model	Ownership	Security	Cost	Scalability
Public	Provider (shared)	Lower	Low	Very high
Private	Single organization	High	High	Limited
Hybrid	Mixed	Custom	Medium	High
Multi-cloud	Multiple providers	Varies	Varies	Very high

Textbook comparison: public clouds are cost-effective but less secure; private clouds are more secure but expensive; hybrid clouds offer flexibility, while multi-clouds provide resilience.

Long Question 2: Classify the various types of cloud services and compare them, highlighting key distinctions.

Cloud services are typically categorized into three main types – IaaS, PaaS and SaaS – each offering different levels of control, flexibility and management.



Figure 6.1: Types of Cloud Services

- **Infrastructure as a Service (IaaS)** – basic computing infrastructure (servers, storage, networking) on a pay-as-you-go basis. Users control the operating systems, applications and storage but not the physical infrastructure. Examples: AWS virtual servers, Microsoft Azure, Google Compute Engine.
- **Platform as a Service (PaaS)** – a complete development and deployment environment in the cloud with infrastructure, middleware, development tools and management services. Developers focus on coding without managing hardware and software layers. Examples: Google App Engine, Microsoft Azure App Services, Heroku.
- **Software as a Service (SaaS)** – software applications hosted and managed by the provider; users subscribe and use them over the internet with no hardware management or software updates. Examples: Google Workspace (Gmail, Docs, Drive), Microsoft Office 365, Salesforce.

Feature	IaaS	PaaS	SaaS
Control	High (OS, apps)	Medium (apps only)	Low (configuration only)
Management	User manages OS/apps	Provider manages the platform	Fully managed by provider
Typical user	System admins, DevOps	Developers	End users
Example	AWS virtual servers	Google App Engine	Gmail

Long Question 3: Discuss the advancements and benefits of edge computing in modern technology.

Edge computing brings processing power closer to data sources, reducing latency and improving efficiency. Instead of relying only on centralized data centers, data is processed at the "edge" of the network, near the source. This minimizes the time data takes to travel, leading to faster decision-making and real-time processing.

Advancements

- **Growth of IoT** – billions of sensors and devices produce too much data to send everything to the cloud.
- **5G networks** – high-speed, low-latency connectivity enables real-time edge processing.
- **AI at the edge** – lightweight models run on devices such as smartphones and cameras.
- **Better edge hardware** – powerful, energy-efficient edge servers and gateways.

Benefits

- **Reduced latency** – data is processed locally.
- **Bandwidth savings** – only important data is sent to the cloud.
- **Better privacy and security** – sensitive data can stay local.
- **Reliability** – works even with unreliable cloud connectivity.
- **Real-time decision making** without a round trip to the cloud.

Examples

- **Autonomous vehicles** – data from sensors and cameras is processed inside the vehicle for quick responses to changing road conditions and better safety (textbook example).
- **Smart cities, healthcare monitoring and industrial automation** – applications needing real-time processing and low latency (textbook tidbit).

Long Question 4: Explain the concept of serverless architectures and their impact on application development.

Serverless architectures allow developers to build and deploy applications without managing servers, improving scalability and reducing operational complexity. In a serverless model the cloud provider automatically allocates resources as needed, and developers pay only for the actual usage of computing resources.

Example (textbook): AWS Lambda is a serverless computing service that lets developers run code without provisioning or managing servers, so they can focus on writing code and building applications rather than managing infrastructure.

How it works

1. The developer writes small pieces of code (functions).
2. The function runs when an event happens, such as a user request or a file upload.
3. The provider automatically provides the needed computing resources and scales them up or down.
4. When the work is done the resources are released, and the developer is charged only for the time used.

Impact on application development

- **No server management** – no need to set up, patch or maintain servers, so developers focus on code.
- **Automatic scalability** – the application handles small or large traffic without manual scaling.
- **Cost savings** – pay only for actual usage; nothing is paid while the code is idle.
- **Faster development and deployment** – small functions can be built and released quickly.
- **Less operational complexity** – the provider handles infrastructure tasks.

Limitations

- Depends on the cloud provider (vendor lock-in).
- Not ideal for long-running or very heavy tasks.
- Debugging and monitoring can be harder than on your own servers.

Long Question 5: Describe "Cloud Deployment Models" with examples.

Cloud deployment models define where the infrastructure resides and who has access to it.

Model	Description	Examples / use case
Public cloud	Owned by a third-party provider and delivered over the internet	AWS, Microsoft Azure, Google Cloud Platform – startups, small businesses, web hosting
Private cloud	Dedicated to a single organization; on-premises or hosted	VMware, OpenStack, IBM Private Cloud – banks, government, healthcare
Hybrid cloud	Public + private clouds working together	AWS Outposts, Azure Stack, Google Anthos – e-commerce keeping customer data private and using public cloud for seasonal traffic spikes
Multi-cloud	Services from multiple public cloud providers	AWS for compute and Azure for AI services – avoids vendor lock-in

Textbook examples: a large bank uses a private cloud for sensitive customer data; a global retail company uses AWS for its e-commerce website, Azure for internal ERP and Google Cloud for analytics.

Chapter 7: Legal and Ethical Aspects of Computing System

Long Question 1: Explain the common clauses found in Terms of Use and describe how they protect both the service provider and the user.

Terms of Use (also called Terms and Conditions or Terms of Service) are legal agreements between a service provider and a user. Common clauses:

Clause	What it says	How it protects
User obligations	What the user must do (give accurate information, not misuse the service)	Provider: protected from misconduct. User: clear expectations.
Limitations of liability	Limits the provider's responsibility if something goes wrong (outage, data loss)	Provider: shielded from excessive lawsuits. User: knows the risks.
Privacy and data use	How personal data is collected, stored and shared	Provider: legal compliance. User: knows their data rights.
Intellectual property rights	Software, logos and content belong to the provider	Provider: prevents unauthorized copying. User: knows what can be used.
Termination of service	When the provider may suspend or end an account	Provider: can remove abusive users. User: forewarned.
Dispute resolution	How conflicts are settled (arbitration, jurisdiction)	Both: avoids costly litigation; users know how to complain.
Modifications to terms	Provider may update the terms; continued use means acceptance	Provider: can adapt to legal/technical change. User: should review updates.

Example from the textbook: when you use a Pakistani online shopping platform like Daraz, the Terms of Use include clauses on payment methods, return policies and delivery procedures, so customers know what to expect and what is expected of them. By clearly stating rights and duties, Terms of Use create a transparent relationship and provide legal safeguards to both sides.

Long Question 2: Explain the concept of bridging the digital divide and discuss the roles of government initiatives in addressing this issue. Provide relevant examples from Pakistan.

What is the digital divide?

The digital divide is the gap between individuals, communities and regions that have access to modern Information and Communication Technology (ICT) and those that do not. It leads to unequal opportunities in education, employment, healthcare and social participation.

Causes (textbook)

- **Economic barriers** – many families cannot afford computers, smartphones or broadband.
- **Geographical barriers** – rural and remote areas often lack internet infrastructure.
- **Educational barriers** – lack of digital literacy and training.
- **Social barriers** – age, gender and disability (e.g. cultural barriers limiting women's access).

Impacts – with Pakistan examples

- **Educational inequality** – during COVID-19, students from low-income and rural families in Pakistan struggled with online learning for lack of internet or devices.
- **Economic disparities** – online job portals and digital banking are mostly available in urban areas.
- **Social and civic participation** – e-government services are more readily available in cities.
- **Health disparities** – rural communities may struggle to find reliable health information or use telemedicine.
- **Digital literacy gap** – people without access miss the chance to learn digital skills.

Bridging the digital divide

Bridging the digital divide means making sure everyone has equal access to technology and the internet, so everyone has the same chances to learn and grow. Ways to do this (textbook):

1. **Government initiatives** – the government builds the infrastructure needed for internet access where it is not available. The Government of Pakistan has launched several initiatives to increase internet access in rural areas, including mobile broadband services in remote regions.
2. **Educational programs** – schools and community centers teach people how to use technology.
3. **Public-private partnerships** – the government teams up with businesses and non-profits to make technology more affordable.

Role of the government

- Build infrastructure (networks, broadband and mobile coverage) in underserved areas.
- Make devices and internet access affordable through policies and partnerships.
- Support digital literacy programs in schools and communities.
- Provide e-government and online services that are accessible to all citizens.

Textbook example: Pakistan's "Internet in a Box" style projects and other initiatives provide digital education resources to remote areas even without internet access, showing how community-based solutions help bridge the divide.

Long Question 3: Explain how critical thinking skills contribute to responsible information utilization. Provide examples of how these skills can be applied in real-life scenarios.

Critical thinking is the ability to analyze information objectively, evaluate sources and make reasoned judgments. It is essential for responsible use of information in the digital age.

- **Source evaluation** – separating credible news from fake news.
- **Bias detection** – recognizing sponsored content and hidden agendas.
- **Fact-checking** – verifying claims with multiple sources.
- **Logical reasoning** – spotting weak or unsupported arguments.

Real-life scenarios

1. **Social media** – before sharing a sensational headline, check the original source, date and author.

2. **Health information** – for a "miracle cure", look for scientific studies and consult medical professionals.
3. **Shopping** – read reviews critically: look for verified purchases and compare platforms; beware of fake reviews.
4. **Academic research** – use peer-reviewed journals rather than random websites and cite sources to avoid plagiarism.

By applying critical thinking, individuals become responsible digital citizens who contribute to an informed society.

Long Question 4: Discuss the importance of collaborative problem solving in a digital environment and provide examples of how human-machine collaboration can enhance this process.

Collaborative problem solving means several people working together, using diverse skills and perspectives, to solve a challenge. Digital technology amplifies this.

Importance

- **Diverse expertise** – coder, designer and marketer bring different knowledge.
- **Faster solutions** – parallel work and instant communication (Slack, Teams).
- **Innovation** – shared brainstorming tools (Miro, Jamboard) produce more creative ideas.
- **Global reach** – teams can collaborate across countries, around the clock.
- **Sharing insights** – exchanging ideas and feedback improves the final result.

Human-machine collaboration examples

- **Data analysis** – machines process large datasets (e.g. Python/Pandas) while humans interpret results and decide strategy.
- **Design** – architects use CAD software while AI suggests optimized layouts, giving faster iterations and fewer errors.
- **Customer service** – chatbots handle routine queries, humans handle complex issues, so support is available 24/7.
- **Healthcare** – AI analyses medical images and doctors make the final diagnosis, improving accuracy and early detection.

Digital collaboration tools combined with machine intelligence make teams more productive and innovative.

Chapter 8: Online Research and Digital Literacy

Long Question 1: Discuss the different types of online research and their purposes, providing examples for each type.

Type	Purpose	Example
General information research	Finding basic facts or overviews on a topic	Searching Google for "history of the internet" for a school essay
Academic research	Scholarly inquiry using books, journals and academic databases	Using Google Scholar or PubMed to collect papers for a thesis
Market research	Understanding customers, competitors and industry trends	A startup surveys social media users to gauge demand for a new app
Fact-checking research	Verifying the accuracy of claims or news	Using Snopes or official government websites to check a viral story
Health research	Finding reliable medical information	Looking up symptoms on a reputable site such as Mayo Clinic before seeing a doctor
Technical / how-to research	Learning to do a task or solve a technical problem	Watching YouTube tutorials to fix a leaking tap
Legal research	Finding laws, regulations and case studies	A lawyer uses online legal databases to prepare a case

Each type needs different sources and different evaluation criteria.

Long Question 2: Explain the concept of digital literacy and its key components. How does it contribute to effective use of digital tools and resources?

Digital literacy is the ability to use digital technologies effectively, critically and safely. It goes beyond basic technical skills to include cognitive and ethical competencies.

Key components

1. **Technical proficiency** – operating devices, using software and browsers, managing files.
2. **Information literacy** – searching, locating, evaluating and organizing digital information; judging credibility.
3. **Communication and collaboration** – email, social media, Google Docs, Slack.
4. **Content creation** – producing documents, videos and websites; understanding copyright and licensing.
5. **Digital safety and privacy** – protecting personal data, recognizing phishing, managing passwords.
6. **Critical thinking** – analyzing online content for bias, misinformation and propaganda.

Contribution to effective use of digital tools

- **Efficiency** – skilled users find information faster and use tools productively.
- **Accuracy** – evaluating sources reduces reliance on false information.
- **Creativity** – digital tools enable creation and innovation.

- **Responsibility** – ethical use, respect for intellectual property and good online etiquette.

Digital literacy empowers people to take part fully in a connected society – in education, employment and civic life.

Long Question 3: Explain the process of developing a clear and focused research question. Illustrate your answer with examples of well-formulated and poorly-formulated research questions.

A research question guides the whole research process. A good one is clear, focused, feasible and debatable.

1. **Choose a broad topic** – e.g. "Climate change".
2. **Narrow it down** – e.g. "Effects of climate change on agriculture".
3. **Make it specific** – add place, population and time: "Impact of rising temperatures on wheat yield in Punjab, Pakistan (2000–2020)".
4. **Ensure it is researchable** – can data be collected and is the scope manageable?
5. **Formulate it as a question** – "How has the increase in average temperature affected wheat production in Punjab from 2000 to 2020?"

Poorly formulated	Well formulated
"What is social media?"	"How does Instagram usage affect body image among teenage girls in Lahore?"
"Is pollution bad?"	"What is the correlation between PM2.5 levels and hospital admissions for asthma in Karachi?"
"Why are people poor?"	"What are the primary barriers to microfinance adoption among rural women in Sindh?"

Characteristics of a good research question (FINER)

Feasible, Interesting, Novel, Ethical, Relevant – it must be doable with available time and resources, engage readers, add new knowledge, cause no harm and matter to science or society.

Long Question 4: Discuss the importance of research ethics in maintaining the credibility and reliability of research findings. Explain how unethical practices could affect the research community and society.

Research ethics are the moral principles that guide honest, fair and responsible research. The textbook lists key principles: informed consent, confidentiality, integrity and respect for participants.

Importance

- **Credibility** – ethical research produces trustworthy results others can build on.
- **Protection of participants** – informed consent, confidentiality and avoidance of harm.
- **Accountability** – researchers are answerable for their work.
- **Public trust** – society relies on research for policy, medicine and technology.

Effects of unethical practices

On the research community	On society
Retractions and wasted resources; damage to institutional reputation; slowed scientific progress due to misinformation	Harm to participants (e.g. the Tuskegee syphilis study); flawed policies based on fabricated data; public distrust of science

Examples of unethical practice: plagiarism, data fabrication, data falsification, not obtaining consent and biased selection of results. Following ethics is both a legal duty and a professional responsibility.

Long Question 5: Discuss the various types of intellectual property and provide examples of each. Explain how each type helps in protecting different kinds of creations and innovations.

Intellectual property (IP) refers to creations of the mind – inventions, literary and artistic works, designs, symbols, names and images used in commerce – protected by law so creators are rewarded for their work.

Type	What it protects	Textbook example	How it helps
Patent	An invention – a product or process giving a new technical solution	A Pakistani engineer's more efficient solar panel	Only the inventor can make, use or sell it for a set period, encouraging innovation
Trademark	Sign, logo or name that distinguishes a company's goods	The logo of National Foods	Helps consumers identify the source; builds brand recognition
Copyright	Literary and artistic works – books, music, films, paintings, software	A Pakistani author's novel	Only the author can publish, sell or adapt it (e.g. into a movie)
Industrial design	Visual design – shape, configuration and appearance of a product	A unique bottle shape for a new mineral water brand from Karachi	Stops others from using the same look
Trade secret	Confidential information that gives a competitive edge	The recipe of the Pakistani soft drink Pakola	Protected by confidentiality agreements rather than disclosure

How to protect intellectual property (textbook)

- **Patents** – apply through the Intellectual Property Organization of Pakistan (IPO Pakistan).
- **Trademarks** – register your brand name, logo or slogan.
- **Copyrights** – register creative works with the appropriate authority.
- **Trade secrets** – keep information confidential and use legal agreements.

Importance: IP protection encourages people to keep producing new ideas because they know they will be rewarded, and it supports creativity and progress.

Chapter 9: Entrepreneurship in Digital Age

Long Question 1: Discuss the importance of Design Thinking in developing business solutions. Provide examples of how it can be applied.

Design Thinking is a human-centred, iterative problem-solving method that focuses on understanding user needs, challenging assumptions and redefining problems. It has five steps (textbook):

1. **Empathise** – understand the people you are solving the problem for.
2. **Define** – clearly state the problem after gathering information.
3. **Ideate** – brainstorm many ideas.
4. **Prototype** – build a simple version of the idea.
5. **Test** – try it with users and improve it.

Importance in business

- **User-centric** – products are built on real user needs, so they are adopted more.
- **Reduces risk** – early prototyping and testing catch flaws before heavy investment.
- **Fosters innovation** and out-of-the-box ideas.
- **Encourages cross-functional collaboration.**
- **Iterative improvement** through continuous feedback.

Examples

- **Airbnb** – the founders found poor listing photos were a barrier; they took professional photos, increasing bookings.
- **Banking app** – empathising with elderly users led to a simple interface with larger fonts and voice commands.
- **Food delivery** – noticing group orders led to a split-payment feature that was prototyped and tested.

Long Question 2: Explain the process of creating a business plan, and why each step is critical to the success of a new business.

A business plan is a written document describing your business idea, how you plan to make it successful and how it will be funded and run. Its key parts:

Step / section	What it contains	Why it is critical
Executive summary	Overview: mission, product, market and financial highlights	Often the only part investors read; must be compelling
Business description	Problem, solution, value proposition, target customers	Clarifies the concept and what makes it different
Market analysis	Industry size, growth, customers, competitors	Shows there is demand and that you understand the market
Products or services	Offerings, features, benefits	Shows what you sell and why it is unique

Step / section	What it contains	Why it is critical
Marketing and sales strategy	Pricing, promotion, channels	Connects the product to revenue
Management team	Founders and key people, expertise	Investors back people; shows ability to execute
Financial plan	Income, cash flow, balance sheet, funding needs	Proves financial viability
Operational plan	Daily logistics, location, technology, suppliers	Shows you can deliver the product
Appendices	Supporting documents (resumes, patents, charts)	Provides evidence for the claims

Skipping any step leaves gaps that can lead to failure. Software tools (for example business plan templates and collaborative tools such as Google Drive or Dropbox) can help build and share a plan.

Long Question 3: How does market insight influence business decisions? Give examples of how businesses can use this information.

Market insight is a deep understanding of customer behaviour, market trends and competitors, obtained by analysing data. It turns intuition into evidence-based strategy.

Area	Insight	Decision
Product development	Customers prefer eco-friendly packaging	Switch to biodegradable materials and promote it
Pricing	A price-sensitive segment exists	Introduce smaller, lower-cost packs
Marketing	Instagram is the main platform of the target age group	Spend the budget on Instagram ads and influencers
Distribution	60% of orders come from one district	Open a local warehouse to cut delivery time
Customer service	Long waiting time is the top complaint	Add a chatbot for instant replies

Examples: Netflix uses viewing data to decide which original series to produce; Daraz analyses shopping behaviour during Ramadan to offer targeted discounts (the textbook notes that many Pakistani businesses increase advertising during Ramadan and Eid); Starbucks introduced oat milk after seeing demand for plant-based milk.

Long Question 4: What are the essential components of a marketing and sales strategy? Discuss how these strategies can be tailored for different markets.

A marketing and sales strategy explains how a company attracts and converts customers. Essential components:

1. **Target market** – the specific customer groups to focus on.
2. **Value proposition** – why customers should buy from you.
3. **Marketing mix (4Ps)** – Product, Price, Place (channels) and Promotion.

4. **Sales strategy** – how leads are converted (direct sales, e-commerce, partnerships).
5. **Budget and KPIs** – money allocated and measures such as ROI, cost of acquiring customers and conversion rate.

Tailoring for different markets

Market	Approach
B2B vs B2C	B2B: longer sales cycles, relationship-based, content such as whitepapers/LinkedIn. B2C: emotional appeals, mass media, social commerce.
Domestic vs international	Domestic: local language, culture and payment methods. International: adapt to local rules, currencies and customs (e.g. halal menus in Pakistan).
Urban vs rural	Urban: digital ads, trendy packaging. Rural: radio, local influencers, small affordable pack sizes.
Generations	Gen Z: TikTok, influencers, sustainability. Older buyers: email, trust-based messages, clear instructions.

Long Question 5: Discuss the role of communication and storytelling in building a brand and connecting with customers.

Communication and storytelling go beyond giving information – they create emotional connections, set the brand apart and build loyalty.

Role of communication (clear, consistent, credible, engaging)

- **Clarity** – clearly say what the brand stands for.
- **Consistency** – keep the same voice on all channels.
- **Credibility** – honest, transparent messages build trust.
- **Engagement** – two-way communication on social media builds a community.

Role of storytelling

- **Emotional connection** – e.g. TOMS Shoes gives a pair of shoes to a child in need for every purchase.
- **Simplifies complex ideas** – e.g. Apple's "1984" commercial.
- **Humanises the brand** – stories of founders, employees and customers; e.g. Nike's ads with real athletes.
- **Differentiation** – e.g. Khaadi tells the story of Pakistani heritage and handwoven fabrics.

How to craft a brand story

1. **Hero** – the customer, not the brand.
2. **Conflict** – the problem the customer faces.
3. **Resolution** – how your product solves it.
4. **Moral** – the values your brand upholds.

Developing communication and storytelling skills (textbook tips)

- Practise regularly and know your audience.
- Use visual aids such as pictures and diagrams.

- Be clear and concise, and show emotion.

Great stories make customers feel part of something bigger; combined with consistent communication they turn transactions into relationships and customers into advocates.