Class 11 Computer Chapter 3: Algorithms and Problem Solving

MULTIPLE CHOICE QUESTIONS (MCQS) - COMPLETE GUIDE

Introduction to Computational Problems

Q1 What is a computational problem?

a) solved using algorithm
b) solved using creativity
c) has unclear input
d) has no process

✓ **Correct Answer:** a) solved using algorithm

Q2 What does "input" mean in a computational problem?

a) result after processing
b) steps to solve
c) data at start
d) type of problem

✓ **Correct Answer:** c) data at start

Q3 Which problem gives a yes/no answer?

a) search
b) optimization
c) decision
d) counting

✓ **Correct Answer:** c) decision

Q4 Which is an ill-defined problem?

a) check even number
b) find shortest path
c) reduce poverty
d) count book arrangements

✓ **Correct Answer:** c) reduce poverty

Q5 **What is output when checking if 8 is even?**

a found at 2
b even
c odd
d yes

✓ **Correct Answer:** b) even

Algorithms for Problem Solving

Q6 **What is role of algorithm in problem solving?**

a gives steps
b ignores logic
c random process
d blocks solutions

✓ **Correct Answer:** a) gives steps

Q7 **What does PageRank algorithm analyze?**

a page links
b page color
c page size
d page font

✓ **Correct Answer:** a) page links

Q8 **Why use real-life tasks in algorithm learning?**

a show steps in life
b make it harder
c reduce interest
d add confusion

✓ **Correct Answer:** a) show steps in life

Q9 **What is Generate and Test algorithm?**

a tests all options
b picks one solution
c skips testing
d uses no logic

✓ **Correct Answer:** a) tests all options

Q10 **When is Generate and Test best?**

a small problem space
b large problem space
c no rules
d uncertain output

✓ **Correct Answer:** a) small problem space

Q11 **The Halting Problem asks if a program will:**

a stop or run forever
b produce output
c run for a fixed time
d use less memory

✓ **Correct Answer:** c) run for a fixed time

Q12 **The Halting Problem is an example of:**

a solvable problem
b tractable problem
c unsolvable problem
d P-class problem

✓ **Correct Answer:** c) unsolvable problem

Solvability and Unsolvability

Q13 **A problem is solvable if:**

a There is a known algorithm
b It requires random guesses
c It has no clear steps
d It cannot be solved by computers

✓ **Correct Answer:** a) There is a known algorithm

Q14 **Unsolvable problems are those that:**

a Can be solved by brute force
b Can be solved by approximation
c Cannot be solved by any algorithm
d Can be solved in polynomial time

✓ **Correct Answer:** c) Cannot be solved by any algorithm

Q15 **What are "tractable" problems?**

a Cannot be solved
b Solvable in polynomial time
c Require exponential time
d Always solvable

✓ **Correct Answer:** b) Solvable in polynomial time

Q16 **Intractable problems are:**

a Easy to solve quickly
b Practical for large inputs
c Not solvable in polynomial time
d Always in class P

✓ **Correct Answer:** c) Not solvable in polynomial time

Q17 **The Travelling Salesman Problem is:**

a P problem
b NP-hard problem
c Tractable
d Always solvable in linear time

✓ **Correct Answer:** b) NP-hard problem

Complexity Classes

Q18 **Problems in class P are solved in:**

a Exponential time
b Polynomial time
c Non-deterministic time
d Infinite time

✓ **Correct Answer:** b) Polynomial time

Q19 **NP problems can be:**

a Solved in polynomial time
b Verified in polynomial time
c Solved in exponential time only
d Not verified

✓ **Correct Answer:** b) Verified in polynomial time

Q20 **NP-hard problems are:**

a Easy to solve
b As hard as NP problems
c In class P
d Always solvable

✓ **Correct Answer:** b) As hard as NP problems

Q21 **Problems in NP and NP-hard are called:**

a P
b BPP
c NP-complete
d Tractable

✓ **Correct Answer:** c) NP-complete

Q22 **Which is an NP-complete problem?**

a Sorting
b Knapsack
c GCD finding
d Binary search

✓ **Correct Answer:** b) Knapsack

Q23 **If P = NP, this means:**

a Some NP unsolvable
b All NP solvable in polynomial time
c All NP unsolvable
d Same space complexity

✓ **Correct Answer:** b) All NP solvable in polynomial time

Q24 **Which is true about P, NP, NP-complete?**

a NP-complete ⊆ P
b P ⊆ NP
c NP = NP-complete
d NP-complete ⊆ P

✓ **Correct Answer:** b) P ⊆ NP

Q25 Why is tractability important?

a Output color
b Language choice
c Algorithm efficiency in practice
d Code size

✓ **Correct Answer:** c) Algorithm efficiency in practice

Q26 Easy to verify but hard to solve problems are in:

a P
b NP
c NP-hard
d Undecidable

✓ **Correct Answer:** b) NP

Q27 The Halting Problem is:

a Solvable & tractable
b Unsolvable & in P
c Unsolvable & not in NP
d NP-complete

✓ **Correct Answer:** c) Unsolvable & not in NP

Q28 Which diagram shows P, NP, NP-hard, NP-complete?

a Bar chart
b Pie chart
c Venn diagram
d Line graph

✓ **Correct Answer:** c) Venn diagram

Q29 NP-complete matters because:

a Solved faster than P
b Solving one solves all NP
c Always tractable
d No real use

✓ **Correct Answer:** b) Solving one solves all NP

Q30 **P vs NP asks if:**

a Fast solutions are in NP
b Checkable = Solvable fast
c Tractable = Unsolvable
d All in P

✓ **Correct Answer:** b) Checkable = Solvable fast

Algorithm Analysis

Q31 **What is the main purpose of algorithm analysis?**

a Write code
b Check complexity
c Debug errors
d Improve hardware

✓ **Correct Answer:** b) Check complexity

Q32 **What does time complexity measure?**

a Memory use
b Time growth with input
c Number of loops
d Output accuracy

✓ **Correct Answer:** b) Time growth with input

Q33 **Which notation shows the upper bound of time complexity?**

a Omega
b Theta
c Big O
d Lambda

✓ **Correct Answer:** c) Big O

Q34 **What does O(1) mean in Big O notation?**

a Linear time
b Constant time
c Quadratic time
d Logarithmic time

✓ **Correct Answer:** b) Constant time

Q35 **What does O(n²) indicate?**

a Linear growth
b Constant time
c Square of input
d Logarithmic growth

✓ **Correct Answer:** c) Square of input

Q36 **Which is an example of O(log n) complexity?**

a Sorting numbers
b Pair comparison
c Binary search
d Linear search

✓ **Correct Answer:** c) Binary search

Q37 **What does space complexity measure?**

a Execution time
b Memory usage
c Number of steps
d Result accuracy

✓ **Correct Answer:** b) Memory usage

Algorithm Design Techniques

Q38 **What is the primary goal of algorithm design techniques?**

a Improve hardware
b Solve problems systematically
c Reduce input size
d Debug software

✓ **Correct Answer:** b) Solve problems systematically

Q39 **How does Divide and Conquer work?**

a Make local choices
b Break, solve, combine
c Store subproblem results
d Try all options, backtrack

✓ **Correct Answer:** b) Break, solve, combine

Q40 **Which technique is used in Merge Sort?**

a Greedy
b Dynamic Programming
c Divide and Conquer
d Backtracking

✓ **Correct Answer:** c) Divide and Conquer

Q41 **What defines a Greedy Algorithm?**

a Explore all solutions
b Make best local choice
c Store results
d Divide problems

✓ **Correct Answer:** b) Make best local choice

Q42 **Which is a Greedy Algorithm example?**

a Fibonacci sequence
b Coin Change
c Merge Sort
d Puzzle solving

✓ **Correct Answer:** b) Coin Change

Q43 **What is a key limitation of Greedy Algorithms?**

a Too complex
b Always optimal
c Not always optimal
d High memory use

✓ **Correct Answer:** c) Not always optimal

Q44 **What is the main advantage of Dynamic Programming?**

a Avoid redundant work
b Locally optimal results
c Solve independent problems
d Explore all options

✓ **Correct Answer:** a) Avoid redundant work

Q45 **Which is solved by Dynamic Programming?**

a Coin Change
b Fibonacci sequence
c Merge Sort
d Puzzle solving

✓ **Correct Answer:** b) Fibonacci sequence

Q46 **How does Backtracking work?**

a Break into parts
b Make local choices
c Build and backtrack
d Store subproblems

✓ **Correct Answer:** c) Build and backtrack

Q47 **For which problems is Backtracking suitable?**

a Optimal substructure
b Explore all combinations
c One optimal choice
d No overlapping problems

✓ **Correct Answer:** b) Explore all combinations

Commonly Used Algorithms

Q48 **Why are commonly used algorithms important?**

a Reduce hardware cost
b Solve common problems efficiently
c Remove need for data structures
d For theory only

✓ **Correct Answer:** b) Solve common problems efficiently

Q49 **What do sorting algorithms mainly do?**

a Network routing
b Arrange data
c Find node links
d Search data

✓ **Correct Answer:** b) Arrange data

Q50 **Which is a sorting algorithm?**

a Linear Search
b DFS
c Quick Sort
d BFS

✓ **Correct Answer:** c) Quick Sort

Q51 **How does Bubble Sort work?**

a Select smallest and swap
b Swap adjacent if wrong
c Divide and merge
d Explore nodes level-wise

✓ **Correct Answer:** b) Swap adjacent if wrong

Q52 **What is the time complexity of Bubble Sort?**

a O(n log n)
b $O(n^2)$
c O(log n)
d O(1)

✓ **Correct Answer:** b) $O(n^2)$

Q53 **Why is Bubble Sort not for large datasets?**

a Needs more memory
b $O(n^2)$ time complexity
c Works on sorted data only
d Can't handle numbers

✓ **Correct Answer:** b) $O(n^2)$ time complexity

Q54 **How does Selection Sort work?**

a Swap adjacent
b Select minimum from unsorted part
c Use queue for nodes
d Store intermediate results

✓ **Correct Answer:** b) Select minimum from unsorted part

Q55 **What is the time complexity of Selection sort?**

a O(n log n)
b O(n^2)
c O(log n)
d O(n)

✓ **Correct Answer:** b) O(n^2)

Q56 **What is the purpose of search algorithms?**

a Arrange data
b Find specific data
c Analyze nodes
d Save memory

✓ **Correct Answer:** b) Find specific data

Q57 **How does Linear Search work?**

a Halve search interval
b Check each item one by one
c Explore branch deeply
d Use queue for nodes

✓ **Correct Answer:** b) Check each item one by one

Q58 **What must Binary Search have?**

a Unsorted data
b Sorted data
c Graph data
d Stack process

✓ **Correct Answer:** b) Sorted data

Q59 **What is the time complexity of Binary Search?**

a O(n^2)
b O(n)
c O(log n)
d O(V + E)

✓ **Correct Answer:** c) O(log n)

Q60 **Which algorithm finds node relationships in a graph?**

a Bubble Sort
b Linear Search
c BFS
d Selection Sort

✓ **Correct Answer:** c) BFS

Q61 **What is the time complexity of BFS and DFS?**

a O(n log n)
b O(V + E)
c $O(n^2)$
d O(log n)

✓ **Correct Answer:** b) O(V + E)

Q62 **How do BFS and DFS differ?**

a BFS-stack, DFS-queue
b BFS-level, DFS-depth
c BFS-sort, DFS-search
d BFS always slower

✓ **Correct Answer:** b) BFS-level, DFS-depth

Answer Key Summary

Q.No.	Answer	Q.No.	Answer	Q.No.	Answer	Q.No.	Answer
1	a	17	b	33	c	49	b
2	c	18	b	34	b	50	c
3	c	19	b	35	c	51	b
4	c	20	b	36	c	52	b
5	b	21	c	37	b	53	b
6	a	22	b	38	b	54	b
7	a	23	b	39	b	55	b
8	a	24	b	40	c	56	b
9	a	25	c	41	b	57	b
10	a	26	b	42	b	58	b
11	c	27	c	43	c	59	c
12	c	28	c	44	a	60	c
13	a	29	b	45	b	61	b
14	c	30	b	46	c	62	b
15	b	31	b	47	b		
16	c	32	b	48	b		

Topic-wise Distribution of MCQs

Topic	Question Numbers	Total Questions
Introduction to Computational Problems	1-5	5
Algorithms for Problem Solving	6-12	7
Solvability and Unsolvability	13-17	5
Complexity Classes	18-30	13
Algorithm Analysis	31-37	7
Algorithm Design Techniques	38-47	10
Commonly Used Algorithms	48-62	15
Total		62

Chapter Summary - Quick Reference

Topic	Key Points
Computational Problem	Input → Process → Output
Problem Types	Decision (yes/no), Search (find item), Optimization (best solution), Counting (number of ways)
Well-defined Problem	Clear goals, inputs, process, output
Ill-defined Problem	Unclear goals, vague inputs, no fixed process
Generate and Test	Generate possible solutions, test each one
Solvable Problems	Can be solved by algorithm in finite time
Unsolvable Problems	Cannot be solved by any algorithm
Tractable Problems	Solvable in polynomial time
Intractable Problems	Require exponential time
Complexity Classes	P (polynomial time), NP (verifiable in polynomial time), NP-hard, NP-Complete
P vs NP	Does P = NP or are they different classes?
Time Complexity	How runtime grows with input size
Space Complexity	How memory usage grows with input size
Big O Notation	O(1), O(log n), O(n), O(n log n), O(n^2), O(2^n), O(n!)
Divide and Conquer	Break, solve, combine (e.g., Merge Sort)
Greedy Algorithms	Make locally optimal choices (e.g., Coin Change)
Dynamic Programming	Store results of overlapping subproblems (e.g., Fibonacci)
Backtracking	Build solution, backtrack if fails (e.g., N-Queens)
Sorting Algorithms	Bubble Sort O(n^2), Selection Sort O(n^2), Quick Sort O(n log n), Merge Sort O(n log n)
Search Algorithms	Linear Search O(n), Binary Search O(log n)
Graph Algorithms	BFS (level-order, finds shortest path), DFS (depth-first, uses less memory)

End of Chapter 3 Multiple Choice Questions