PRACTICE MCQs

Q1. What is recursion in C++?

1. A loop construct used for repetitive tasks.
2. A function calling itself directly or indirectly.
3. A data type used for storing recursive values.
4. A library for managing recursive data structures.

Ans. 2

Q2. Which of the following is NOT a base case in a recursive function?

1. A condition that terminates the recursion.
2. A condition that triggers the recursive calls.
3. The lowest level of recursion where no further recursive calls are made.
4. A condition that prevents the function from running forever.

Ans 2

Q3. What is the purpose of a recursive function's base case?

1. To make the function run faster. To reduce the memory usage of the function.
2. To provide a starting point for the recursion and prevent infinite loops.
3. To increase the complexity of the recursive algorithm.
4. To allow for multiple recursive calls.

Ans 2

Q4. Which of the following is an example of a problem that is well-suited for recursion in C++?

1. Sorting a list of numbers.
2. Calculating the factorial of a number.
3. Adding two numbers together.
4. Printing "Hello, World!" to the console.

Ans 2

Q5. In a recursive function, when does the backtracking phase occur?

1. Before the function makes the first recursive call.
2. After the function reaches the base case and starts returning.
3. At the beginning of the function before any calculations.
4. In the middle of the recursive calls.

Ans 2

Q6. What is the order of execution in a recursive function when multiple recursive calls are made?

1. It always follows a post-order traversal.
2. it always follows a pre-order traversal.
3. It depends on the specific implementation and problem.

4. It follows an in-order traversal.

Ans 3

Q7. Which of the following statements about recursion is true?

1. Recursion is always more memory-efficient than iterative solutions.
2. All recursive problems can be solved iteratively.
3. Recursion is a technique used only in programming languages like C++.
4. Recursive functions can have an unlimited number of base cases.

Ans 2

Q8. In a recursive function, what happens during each recursive call?

1. The function reverts to the previous call's state.
2. The function executes a loop instead of making further calls.
3. The function creates a new stack frame with its own set of variables.
4. The function always returns to the base case.

Ans 3

Q9. Which of the following best describes the concept of "tail recursion" in C++?

1. A type of recursion that involves recursion within loops.
2. A recursion technique that avoids the use of base cases.
3. A recursive function where the recursive call is the last operation in the function.
4. A recursion approach that uses a separate data structure to store intermediate results.

Ans 3

Q10. Which type of recursion occurs when a function calls itself either directly or indirectly, potentially leading to a stack overflow?

1. Direct Recursion
2. Tail Recursion
3. Non-Tail Recursion
4. Indirect Recursion

Ans 4

1. What does STL stand for in C++?
   A) System Template Library
   B) Standard Template Library
   C) Structured Template Library
   D) Static Template Library

   Answer: B) Standard Template Library
2. Which header file is used to include vectors in C++ STL?
   A) <list>
   B) <vector>
   C) <map>
   D) <stack>

Answer: B) <vector>

3. Which STL container is used to implement a Last-In-First-Out (LIFO) data structure?
    A) set
    B) map
    C) stack
    D) queue

Answer: C) stack

4. In STL, which algorithm is used to find the maximum element in a container?
    A) find_max()
    B) max_element()
    C) maximum()
    D) find_maximum()

Answer: B) max_element()

5. What does the 'push_back()' function do in a vector container of the STL?
    A) Adds an element to the front of the vector
    B) Adds an element to the back of the vector
    C) Removes an element from the front of the vector
    D) Removes an element from the back of the vector

Answer: B) Adds an element to the back of the vector

6. Which STL container automatically sorts its elements when a new element is added?
    A) map
    B) unordered_map
    C) set
    D) unordered_set

Answer: C) set

7. What does the 'find()' algorithm in STL do?
    A) Finds the first occurrence of an element in a container
    B) Finds the last occurrence of an element in a container
    C) Finds all occurrences of an element in a container
    D) Finds the index of an element in a container

Answer: A) Finds the first occurrence of an element in a container

8. Which STL container allows only unique elements?
    A) set
    B) unordered_set
    C) vector
    D) list

Answer: A) set

9. In STL, which algorithm is used to sort elements in a container?
    A) order()
    B) sort()
    C) arrange()
    D) organize()

Answer: B) sort()

10. What does the 'pop()' function do in a stack container of the STL?
    A) Adds an element to the stack
    B) Removes the top element from the stack
    C) Removes the bottom element from the stack

D) Retrieves the top element from the stack without removing it

Answer: B) Removes the top element from the stack

Question 1: What is the index of the first element in an array in C++?

a) 1
b) 0
c) -1
d) It depends on the size of the array

Answer: b) 0

Question 2: How do you access the fifth element in an array named arr?

a) arr[4]
b) arr[5]
c) arr[0]
d) arr[1]

Answer: a) arr[4]

Question 3: In C++, how can you find the number of elements in an array named numbers?

a) numbers.length()
b) sizeof(numbers)
c) numbers.size()
d) sizeof(numbers) / sizeof(numbers[0])

Answer: d) sizeof(numbers) / sizeof(numbers[0])

Question 4: What is the purpose of the memset function in C++?

a) To calculate the size of an array
b) To set all elements of an array to a specific value
c) To reverse the elements of an array
d) To find the maximum element in an array

Answer: b) To set all elements of an array to a specific value

Question 5: In C++, what is the correct way to declare a string?

a) string name = "John";
b) char name[] = "John";
c) char name[5] = "John";
d) String name = "John";

Answer: a) string name = "John";

Question 6: How do you find the length of a C-style string (char array) in C++?

a) Using strlen() function
b) Using length() function
c) Using size() function
d) Using sizeof() operator

Answer: a) Using strlen() function

Question 7: Which function is used to concatenate two C-style strings?

a) strcat()
b) concat()
c) append()
d) join()

Answer: a) strcat()

Question 8: What is the correct way to compare two C-style strings in C++?

a) strcompare()
b) strcmp()
c) compare()
d) stringCompare()

Answer: b) strcmp()

Question 9: Which of the following statements is true about arrays in C++?

a) Arrays can only store elements of the same data type.
b) Arrays can dynamically resize during runtime.
c) The size of an array must be known at compile time.
d) Arrays are not supported in C++.

Answer: c) The size of an array must be known at compile time.

Question 10: How do you initialize an array in C++?

a) array = {1, 2, 3};
b) int array[3] = {1, 2, 3};
c) array(1, 2, 3);
d) array = new int[3];

Answer: b) int array[3] = {1, 2, 3};

Question 12: What does the sizeof operator in C++ return for an array?

a) The number of elements in the array.
b) The sum of all elements in the array.
c) The total size (in bytes) of the array.
d) The average value of elements in the array.

Answer: c) The total size (in bytes) of the array.

Question 13: What is the correct way to declare and initialize a character array in C++?

a) char name = "John";
b) char name[] = "John";
c) string name = "John";
d) string name[] = {'J', 'o', 'h', 'n'};

Answer: b) char name[] = "John";

Question 14: How do you find the length of a C++ string object?

a) length() method
b) size() method
c) strlen() function
d) sizeOf() function

Answer: b) size() method

Question 15: Which function is used to copy one C-style string into another?

a) strcpy()
b) copy()
c) strncpy()
d) stringCopy()

Answer: a) strcpy()

Question 16: What is the purpose of the getline() function in C++ when used with strings?

a) To read a line of text from the console.
b) To concatenate two strings.
c) To find the length of a string.
d) To compare two strings.

Answer: a) To read a line of text from the console.

Question 1: What does time complexity of O(n^2) represent for an algorithm?

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

Answer: b) Quadratic time

Question 2: If an algorithm has a time complexity of O(1), what does it imply?

a) It runs in constant time.
b) It runs in logarithmic time.
c) It runs in linear time.
d) It runs in polynomial time.

Answer: a) It runs in constant time.

Question 3: What is the time complexity of a linear search algorithm in an array of size n?

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

Answer: b) O(n)

Question 4: If an algorithm has a time complexity of O(log n), what type of algorithm is it likely to be?

a) Linear algorithm
b) Quadratic algorithm
c) Exponential algorithm
d) Logarithmic algorithm

Answer: d) Logarithmic algorithm

Space Complexity MCQs:

Question 5: What does space complexity of O(1) represent for an algorithm?

a) Constant space
b) Linear space
c) Quadratic space
d) Logarithmic space

Answer: a) Constant space

Question 6: If an algorithm uses additional memory proportional to the input size, what is its space complexity?

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

Answer: b) O(n)

Question 7: What is the space complexity of a recursive algorithm with a depth of recursion log n?

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

Answer: c) O(log n)

Question 8: If an algorithm creates a matrix of size n x n, what is its space complexity?

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

Answer: c) O(n^2)

Question 1: In binary search, what is the key requirement for the array?

a) It must be sorted in descending order.
b) It must be sorted in ascending order.
c) It can be in any order.
d) It must have only unique elements.

Answer: b) It must be sorted in ascending order.

Question 2: What is the time complexity of binary search on a sorted array of size n?

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

Answer: b) O(log n)

Question 3: If the target element is not present in the array during binary search, what does the algorithm return?

a) -1
b) 0
c) The index of the last element
d) An error message

Answer: a) -1

Question 4: Which of the following is a disadvantage of recursive binary search?

a) Simplicity of implementation
b) Stack overflow for large arrays
c) Efficient for unsorted arrays
d) Suitable for dynamic arrays only

Answer: b) Stack overflow for large arrays

Question 5: What is the formula to calculate the mid-point index in binary search?

a) mid = (low + high) / 2
b) mid = (low - high) / 2
c) mid = low + (high - low) / 2
d) mid = low * high

Answer: c) mid = low + (high - low) / 2

Question 6: Which algorithm is often used as a base for binary search?

a) Linear Search
b) Bubble Sort
c) Quick Sort
d) Merge Sort

Answer: d) Merge Sort

Question 7: In binary search, what is the purpose of checking the middle element against the target?

a) To find the index of the target element.
b) To determine if the array is sorted.
c) To decide whether to search the left or right subarray.
d) To reorder the array.

Answer: c) To decide whether to search the left or right subarray.

Question 8: What is the advantage of binary search over linear search?

a) Binary search works for unsorted arrays.
b) Binary search has a lower time complexity.
c) Binary search is easier to implement.
d) Binary search works only for small arrays.

Answer: b) Binary search has a lower time complexity.

Question 1: What is bitmasking commonly used for in programming?

a) Text processing
b) Mathematical calculations
c) Manipulating individual bits in variables
d) String manipulation

Answer: c) Manipulating individual bits in variables

Question 2: In bitwise AND operation (&), what is the result if both bits are 1?

a) 0
b) 1
c) -1
d) No change

Answer: b) 1

Question 3: What is the result of the expression 1 << 3?

a) 8
b) 4
c) 2
d) 1

Answer: a) 8

Question 4: Which bitwise operation is used to set a particular bit to 1?

a) AND (&)
b) OR (|)
c) XOR (^)
d) NOT (~)

Answer: b) OR (|)

Question 5: In bitwise XOR operation (^), what is the result if both bits are the same?

a) 0
b) 1
c) -1
d) No change

Answer: a) 0

Question 6: What is the purpose of the << operator in bitmasking?

a) Left shift
b) Right shift
c) Bitwise AND
d) Bitwise OR

Answer: a) Left shift

Question 7: Which bitwise operation is used to toggle a particular bit?

a) AND (&)
b) OR (|)
c) XOR (^)
d) NOT (~)

Answer: c) XOR (^)

Question 8: What is the result of the expression 5 & 3?

a) 0
b) 1
c) 3
d) 5

Answer: c) 3

Question 1: What is backtracking?

a) A search algorithm
b) An optimization technique
c) A technique to explore all possible solutions
d) A data structure

Answer: c) A technique to explore all possible solutions

Question 2: In backtracking, when is pruning typically done?

a) Before exploring any solution
b) After exploring all solutions
c) During the exploration of solutions
d) Pruning is not used in backtracking

Answer: c) During the exploration of solutions

Question 3: What is the key characteristic of a problem suitable for backtracking?

a) The problem has a unique solution.
b) The problem can be broken into subproblems.
c) The problem is solved using a loop.
d) The problem involves sorting.

Answer: b) The problem can be broken into subproblems.

Question 4: What is the role of the "choice space" in backtracking?

a) It represents all possible choices at each decision point.
b) It limits the number of choices available.
c) It is not relevant in backtracking.
d) It is the final solution space.

Answer: a) It represents all possible choices at each decision point.

Question 5: Which of the following problems is well-suited for backtracking?

a) Linear search
b) Sorting an array
c) Sudoku solving
d) Binary search

Answer: c) Sudoku solving

Question 6: What does the term "backtrack" refer to in the context of backtracking algorithms?

a) Going backward in time
b) Undoing the last decision and trying a different one
c) Iterating over all choices before making a decision
d) Repeating the same decision

Answer: b) Undoing the last decision and trying a different one

Question 7: In backtracking, what is a "partial solution"?

a) The final solution to the problem.
b) An incomplete solution that may be expanded further.
c) The first solution generated during exploration.
d) A solution obtained by pruning choices.

Answer: b) An incomplete solution that may be expanded further.

Question 8: Which data structure is commonly used for keeping track of choices in backtracking?

a) Stack
b) Queue
c) Linked list
d) Array

Answer: a) Stack

Asymptotic Notation MCQs:

Question 9: What does O(n) represent in asymptotic notation?

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

Answer: b) Linear time complexity

Question 10: If an algorithm has a time complexity of O(n^2), how does the running time grow with the size of the input?

a) Linearly
b) Quadratically
c) Exponentially
d) Logarithmically

Answer: b) Quadratically

Question 11: What is the purpose of Big-O notation in computer science?

a) To represent the worst-case time complexity of an algorithm.
b) To represent the average-case time complexity of an algorithm.
c) To represent the best-case time complexity of an algorithm.
d) To represent the space complexity of an algorithm.

Answer: a) To represent the worst-case time complexity of an algorithm.

Question 12: What does Ω(n log n) represent in asymptotic notation?

a) Best-case time complexity
b) Average-case time complexity
c) Worst-case time complexity
d) Lower bound time complexity

Answer: d) Lower bound time complexity

Question 13: If an algorithm has a time complexity of O(1), what can be said about its efficiency?

a) It is very efficient.
b) It is moderately efficient.
c) It is less efficient.
d) It depends on other factors.

Answer: a) It is very efficient.