CS301 Data Structures

Important mcqs

Lec 1 - Introduction to Data Structures

- 1. Which data structure follows the ''last-in-first-out'' (LIFO) principle? a. Queue b. Stack c. Linked List d. Tree Solution: b
- 2. Which data structure allows for efficient insertion and deletion operations in the middle? a. Array b. Stack c. Queue d. Linked List Solution: d
- 3. Which data structure is used to represent hierarchical relationships? a. Array b. Stack c. Queue d. Tree Solution: d
- 4. Which data structure is used to find the shortest path between two nodes in a network? a. Array b. Stack c. Queue d. Graph Solution: d
- 5. Which data structure stores elements in a non-linear and hierarchical manner? a. Array b. Stack c. Queue d. Tree Solution: d
- 6. Which data structure follows the "first-in-first-out" (FIFO) principle? a. Queue b. Stack c. Linked List d. Tree Solution: a
- 7. Which data structure is used to reverse the order of elements in a sequence? a. Array b. Stack c. Queue d. Linked List Solution: b
- 8. Which data structure is used to implement a symbol table? a. Array b. Stack c. Queue d. Hash table Solution: d
- 9. Which data structure is used to sort elements in a sequence? a. Array b. Stack c. Queue d. Heap Solution: d
- 10. Which data structure is used to store and access elements based on a key-value pair? a. Array b. Stack c. Queue d. Dictionary Solution: d

Lec 2 - List Implementation

- 1. Which data structure is used to implement a list?
 - a) Array
 - b) Stack
 - c) Queue
 - d) Tree

Answer: a) Array

- 2. Which of the following is a disadvantage of array-based lists?
 - a) Efficient random access to elements
 - b) Dynamic resizing of the list
 - c) Inefficient insertion and deletion operations
 - d) No need to allocate memory in advance

Answer: c) Inefficient insertion and deletion operations

- 3. Which of the following is a disadvantage of linked-list based lists?
 - a) Efficient random access to elements
 - b) Dynamic resizing of the list
 - c) Inefficient insertion and deletion operations
 - d) No need to allocate memory in advance

Answer: a) Efficient random access to elements

- 4. Which of the following is true for an empty list?
 - a) It has no elements
 - b) It has one element
 - c) It has multiple elements
 - d) None of the above

Answer: a) It has no elements

- 5. Which operation adds an element at the end of an array-based list?
 - a) push()
 - b) pop()
 - c) insert()
 - d) append()

Answer: d) append()

- 6. Which operation adds an element at the beginning of a linked-list based list?
 - a) push()
 - b) pop()
 - c) insert()
 - d) append()

Answer: a) push()

- 7. Which operation removes the last element from an array-based list?
 - a) push()
 - b) pop()
 - c) insert()
 - d) append()

Answer: b) pop()

8. Which operation removes the first element from a linked-list based list?

- a) push()
- b) pop()
- c) insert()
- d) append()

Answer: b) pop()

9. Which of the following is a common application of lists?

- a) Implementing a queue
- b) Implementing a hash table
- c) Implementing a binary search tree
- d) Implementing a heap

Answer: a) Implementing a queue

10. Which of the following is not a type of list?

- a) Singly linked list
- b) Doubly linked list
- c) Circular linked list
- d) Heap linked list

Answer: d) Heap linked list

Lec 3 - Linked List inside Computer Memory

1. In a linked list, each node contains:

- a. A value and a pointer to the previous node
- b. A value and a pointer to the next node
- c. A key and a value
- d. A key and a pointer to the next node

Answer: b

2. The first node in a linked list is called the:

- a. Head
- b. Tail
- c. Root
- d. Leaf

Answer: a

3. In computer memory, each node in a linked list is typically represented as:

- a. A block of memory that contains the value and a pointer to the previous node
- b. A block of memory that contains the value and a pointer to the next node
- c. A hash table that contains the key and the value
- d. An array that contains the key and a pointer to the next node

Answer: b

4. What is the time complexity of inserting a node at the beginning of a linked list?

- a. O(1)
- b. O(n)
- c. O(log n)
- d. O(n log n)

Answer: a

5. What is the time complexity of inserting a node at the end of a linked list?

- a. O(1)
- b. O(n)
- c. O(log n)
- d. O(n log n)

Answer: b

6. Deleting a node from a linked list requires updating the:

- a. Previous node's pointer to the next node
- b. Next node's pointer to the previous node
- c. Current node's value to NULL
- d. None of the above

Answer: a

7. Traversing a linked list means:

- a. Deleting a node from the list
- b. Inserting a node into the list
- c. Moving through the list from the head to the tail
- d. Sorting the list in ascending order

Answer: c

8. Which of the following is a disadvantage of linked lists compared to arrays?

- a. Linked lists allow for efficient insertion and deletion of nodes
- b. Linked lists use memory flexibly
- c. Linked lists can grow dynamically
- d. Linked lists have slow access times for specific nodes

Answer: d

9. Which of the following operations can be performed in constant time on a linked list?

- a. Finding the maximum value in the list
- b. Inserting a node at the end of the list
- c. Removing the head node from the list
- d. Sorting the list in descending order

Answer: c

10. What is the space complexity of a linked list?

- a. O(n)
- b. O(log n)
- c. O(1)
- d. O(n log n)

Answer: a

Lec 4 - Methods of Linked List

- 1. What is the time complexity of inserting an element at the beginning of a singly linked list?
 - a) O(n)
 - b) O(log n)
 - c) O(1)
 - d) O(n^2)

Answer: c) O(1)

- 2. Which of the following is not a type of Linked List?
 - a) Singly Linked List
 - b) Doubly Linked List
 - c) Circular Linked List
 - d) Binary Linked List

Answer: d) Binary Linked List

- 3. What is the time complexity of inserting an element at the end of a singly linked list?
 - a) O(n)
 - b) O(log n)
 - c) O(1)
 - d) O(n^2)

Answer: a) O(n)

- 4. Which of the following is not a pointer used in Linked List?
 - a) head pointer
 - b) tail pointer
 - c) node pointer
 - d) root pointer

Answer: d) root pointer

- 5. What is the time complexity of deleting an element from a singly linked list?
 - a) O(n)
 - b) O(log n)
 - c) O(1)
 - d) O(n^2)

Answer: a) O(n)

- 6. Which of the following Linked List traversal technique involves recursion?
 - a) Linear traversal
 - b) Binary traversal
 - c) Depth First Traversal
 - d) Breadth First Traversal

Answer: c) Depth First Traversal

- 7. Which of the following is a disadvantage of using a doubly linked list?
 - a) Faster traversal in both directions
 - b) Requires more memory than singly linked list
 - c) More difficult to implement than singly linked list
 - d) Allows insertion and deletion of elements only at the beginning of the list

Answer: b) Requires more memory than singly linked list

- 8. Which of the following is not a type of node used in Linked List?
 - a) data node
 - b) header node
 - c) sentinel node
 - d) tail node

Answer: a) data node

- 9. Which of the following Linked List method is used to reverse the order of elements in the list?
 - a) reverse()
 - b) rotate()
 - c) shuffle()
 - d) sort()

Answer: a) reverse()

- 10. Which of the following is a type of circular linked list?
 - a) Circular doubly linked list
 - b) Circular binary linked list
 - c) Circular balanced linked list
 - d) Circular heap linked list

Answer: a) Circular doubly linked list

Lec 5 - Benefits of using circular list

1. What is a benefit of using a circular linked list?

- a) It has a fixed size
- b) It allows for efficient implementation of circular structures
- c) It cannot be traversed multiple times
- d) It is less efficient than a linear linked list

Answer: b

2. What is a circular buffer?

- a) A data structure that can only be accessed in a circular order
- b) A data structure that can only be accessed in a linear order
- c) A buffer that can be efficiently implemented using a circular linked list
- d) A buffer that can only hold a fixed number of elements

Answer: c

3. How can a circular linked list simplify insertion at the beginning or end?

- a) It requires more memory than a linear linked list
- b) It requires less memory than a linear linked list
- c) It requires the same amount of memory as a linear linked list
- d) It cannot simplify insertion at the beginning or end

Answer: b

4. What is a disadvantage of using a circular linked list?

- a) It is less efficient than a linear linked list
- b) It cannot represent circular structures
- c) It is more difficult to implement than a linear linked list
- d) It has a fixed size

Answer: a

5. Which algorithm can be efficiently implemented using a circular linked list?

- a) Binary search
- b) Depth-first search
- c) Breadth-first search
- d) Traversing the list multiple times

Answer: d

6. What is a circular linked list?

- a) A linked list where each node has a pointer to the previous node
- b) A linked list where each node has a pointer to the next node and the previous node
- c) A linked list where the last node points to the first node
- d) A linked list where the first node points to the last node

Answer: c

7. What is a benefit of using a circular linked list for representing a clock?

- a) It is less efficient than a linear linked list
- b) It allows for efficient implementation of circular structures
- c) It requires more memory than a linear linked list
- d) It cannot represent circular structures

Answer: b

8. What is a circular linked list used for?

- a) Representing trees
- b) Implementing binary search
- c) Implementing circular buffering
- d) Storing fixed-size data

Answer: c

9. Can a circular linked list be traversed multiple times?

- a) Yes, but it is less efficient than a linear linked list
- b) No, it can only be traversed once
- c) Yes, and it is more efficient than a linear linked list
- d) Yes, but it requires more memory than a linear linked list

Answer: c

10. How is deletion at the end of a circular linked list implemented?

- a) The last node's pointer is set to NULL
- b) The last node's pointer is set to the first node
- c) The second to last node's pointer is set to NULL
- d) The second to last node's pointer is set to the first node

Answer: c

Lec 6 - Stack From the Previous Lecture

- 1. Which of the following is NOT a characteristic of a stack?
 - a. Follows LIFO principle
 - b. Has two main operations: push and pop
 - c. Can only be implemented using arrays
 - d. Topmost element is the last one added

Answer: c. Can only be implemented using arrays

- 2. What is the time complexity of push and pop operations in a stack implemented using an array?
 - a. O(1)
 - b. O(log n)
 - c. O(n)
 - d. O(n^2)

Answer: a. O(1)

- 3. Which data structure is often used to implement a stack?
 - a. Array
 - b. Linked List
 - c. Queue
 - d. Binary Tree

Answer: b. Linked List

- 4. Which of the following is NOT a common application of stacks?
 - a. Reversing a string
 - b. Evaluating postfix expressions
 - c. Implementing depth-first search in a graph
 - d. Sorting an array

Answer: d. Sorting an array

- 5. Which operation in a stack does not modify the stack?
 - a. Push
 - b. Pop
 - c. Peek
 - d. Size

Answer: c. Peek

- 6. What happens when we try to pop an element from an empty stack?
 - a. The program crashes
 - b. An error message is displayed
 - c. The topmost element becomes NULL
 - d. Nothing happens

Answer: b. An error message is displayed

- 7. Which of the following is NOT a disadvantage of using an array to implement a stack?
 - a. Fixed size
 - b. Elements must be contiguous in memory
 - c. Dynamic resizing is difficult
 - d. Push operation is slower than pop operation

Answer: d. Push operation is slower than pop operation

- 8. What is the maximum number of elements a stack implemented using an array can hold if its size is n?
 - a. n
 - b. n-1
 - c. 2n
 - d. 2n-1

Answer: b. n-1

- 9. Which of the following is NOT a potential application of stacks in computer science?
 - a. Function calls and return values
 - b. Undo and redo operations
 - c. Implementing breadth-first search in a graph
 - d. Checking for balanced parentheses in an expression

Answer: c. Implementing breadth-first search in a graph

- 10. What is the time complexity of searching for an element in a stack implemented using a linked list?
 - a. O(1)
 - b. O(log n)
 - c. O(n)
 - d. O(n^2)

Answer: c. O(n)

Lec 7 - Evaluating postfix expressions

- 1. What is postfix notation?
 - a) Operators are written before their operands
 - b) Operators are written after their operands
 - c) Operators are written in between their operands
 - d) None of the above

Answer: b) Operators are written after their operands

- 2. Which data structure is used to evaluate postfix expressions?
 - a) Queue
 - b) Linked List
 - c) Stack
 - d) Tree

Answer: c) Stack

- 3. Which of the following is an example of a postfix expression?
 - a) 2 + 3
 - b) 2 * 3 + 4
 - c) 23 +
 - d) (2 + 3) * 4

Answer: c) 2 3 +

- 4. What is the first step in evaluating a postfix expression?
 - a) Scan the expression from right to left
 - b) Scan the expression from left to right
 - c) Push the first operand onto the stack
 - d) Pop the first operand from the stack

Answer: b) Scan the expression from left to right

- 5. What happens when an operand is encountered in a postfix expression?
 - a) It is pushed onto the stack
 - b) It is popped from the stack
 - c) It is ignored
 - d) None of the above

Answer: a) It is pushed onto the stack

- 6. What happens when an operator is encountered in a postfix expression?
 - a) It is pushed onto the stack
 - b) It is popped from the stack
 - c) The top two operands are popped from the stack, and the operation is performed on them
 - d) None of the above

Answer: c) The top two operands are popped from the stack, and the operation is performed on them

- 7. Which of the following is an example of a postfix expression that involves multiplication?
 - a) 2 + 3
 - b) 2 * 3 + 4
 - c) 23 *
 - d) (2 + 3) * 4

Answer: c) 2 3 *

- 8. What happens when the entire postfix expression has been scanned?
 - a) The result is the top element in the stack
 - b) The result is the bottom element in the stack
 - c) The stack is empty
 - d) None of the above

Answer: a) The result is the top element in the stack

- 9. Which of the following is an example of a postfix expression that involves subtraction?
 - a) 2 + 3
 - b) 2 * 3 + 4
 - c) 23-
 - d) (2 + 3) * 4

Answer: c) 2 3 -

- 10. Which of the following is an example of a postfix expression that involves division?
 - a) 2 + 3
 - b) 2 * 3 + 4
 - c) 23/
 - d) (2 + 3) * 4

Answer: c) 23/

Lec 8 - Conversion from infix to postfix

- 1. Which of the following data structures is commonly used for converting infix expressions to postfix expressions?
 - a. Stack
 - b. Queue
 - c. Linked List
 - d. Heap

Answer: a. Stack

- 2. In infix notation, where are operators written in relation to their operands?
 - a. Before
 - b. After
 - c. Between
 - d. Both a and c

Answer: d. Both a and c

- 3. What is the first step in converting an infix expression to postfix notation?
 - a. Scan the expression from left to right
 - b. Initialize an empty stack and postfix expression
 - c. Check for balanced parentheses
 - d. Add operands to the postfix expression

Answer: b. Initialize an empty stack and postfix expression

- 4. Which of the following operators has the highest precedence in mathematical expressions?
 - a. Multiplication
 - b. Division
 - c. Addition
 - d. Subtraction

Answer: a. Multiplication

- 5. What is the role of a stack in converting infix to postfix notation?
 - a. To keep track of operators and their precedence levels
 - b. To keep track of operands and their positions
 - c. To perform the necessary calculations
 - d. To check for errors in the expression

Answer: a. To keep track of operators and their precedence levels

- 6. What happens to a left parenthesis when converting from infix to postfix notation?
 - a. It is added to the postfix expression
 - b. It is pushed onto the stack
 - c. It is discarded
 - d. It is popped off the stack

Answer: b. It is pushed onto the stack

- 7. How can errors be handled while converting infix to postfix notation?
 - a. By checking for balanced parentheses
 - b. By checking for errors during scanning
 - c. By using a queue data structure
 - d. Both a and b

Answer: d. Both a and b

8. Which of the following expressions is equivalent to "a + b * c" in postfix notation?

- a. "a b c * +"
- b. "a b + c *"
- c. "a + b c *"
- d. "a b c + *"

Answer: a. "a b c * +"

9. What is the final step in converting an infix expression to postfix notation?

- a. Pop any remaining operators off the stack and add them to the postfix expression
- b. Add operands to the postfix expression
- c. Discard the left parenthesis
- d. Push the right parenthesis onto the stack

Answer: a. Pop any remaining operators off the stack and add them to the postfix expression

10. Which of the following is an advantage of postfix notation over infix notation?

- a. It is easier to read and write
- b. It eliminates the need for parentheses to indicate the order of operations
- c. It is more commonly used in mathematical expressions
- d. Both a and b

Answer: b. It eliminates the need for parentheses to indicate the order of operations.

Lec 9 - Memory Organization

- 1. Which of the following is the fastest type of memory?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: a) Registers

- 2. Which level of memory is located close to the CPU and used to store frequently accessed data?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: b) Cache

- 3. Which level of memory is used to store data and program instructions during execution?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: c) Main memory

- 4. Which type of memory has the largest capacity?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: d) Secondary storage

- 5. Which type of storage is used for long-term data storage?
 - a) Main memory
 - b) Secondary storage
 - c) Cache
 - d) Registers

Answer: b) Secondary storage

- 6. Which level of memory has the slowest access time?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: d) Secondary storage

- 7. Which level of memory is the most expensive?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: a) Registers

- 8. Which level of memory has the largest capacity?
 - a) Registers
 - b) Cache
 - c) Main memory
 - d) Secondary storage

Answer: d) Secondary storage

- 9. Which type of memory organization is used to minimize the time spent accessing data from higher levels of the memory hierarchy?
 - a) Top-down organization
 - b) Bottom-up organization
 - c) Parallel organization
 - d) Hierarchical organization

Answer: d) Hierarchical organization

- 10. Which type of computing system may use different types of memory organization depending on its intended use and performance requirements?
 - a) Embedded systems
 - b) Personal computers
 - c) Supercomputers
 - d) All of the above

Answer: d) All of the above

Lec 10 - Queues

- 1. What is a queue data structure?
 - a) A data structure where the last element added is the first one to be removed.
 - b) A data structure where the first element added is the first one to be removed.
 - c) A data structure where the middle element is always removed first.
 - d) None of the above.

Answer: b) A data structure where the first element added is the first one to be removed.

- 2. Which operation adds an element to the queue?
 - a) Dequeue
 - b) Enqueue
 - c) Peek
 - d) None of the above.

Answer: b) Enqueue

- 3. Which operation removes an element from the queue?
 - a) Dequeue
 - b) Enqueue
 - c) Peek
 - d) None of the above.

Answer: a) Dequeue

- 4. Which operation returns the element at the front of the queue without removing it?
 - a) Dequeue
 - b) Enqueue
 - c) Peek
 - d) None of the above.

Answer: c) Peek

- 5. Which data structure is commonly used to implement a queue?
 - a) Array
 - b) Linked list
 - c) Both a and b
 - d) None of the above.

Answer: c) Both a and b

- 6. Which of the following is not a real-world application of queues?
 - a) Waiting lines at banks
 - b) Amusement parks
 - c) Airports
 - d) None of the above.

Answer: d) None of the above.

- 7. What is the time complexity of the enqueue operation in a queue implemented using an array?
 - a) O(1)
 - b) O(n)
 - c) O(log n)
 - d) None of the above.

Answer: a) O(1)

- 8. What is the time complexity of the dequeue operation in a queue implemented using a linked list?
 - a) O(1)
 - b) O(n)
 - c) O(log n)
 - d) None of the above.

Answer: a) O(1)

- 9. Which of the following is a disadvantage of using an array to implement a queue?
 - a) Insertion and deletion are faster than in a linked list.
 - b) The size of the array must be fixed.
 - c) It is more efficient in terms of memory usage.
 - d) None of the above.

Answer: b) The size of the array must be fixed.

- 10. Which of the following is a disadvantage of using a linked list to implement a queue?
 - a) Insertion and deletion are slower than in an array.
 - b) The size of the linked list must be fixed.
 - c) It is less efficient in terms of memory usage.
 - d) None of the above.

Answer: a) Insertion and deletion are slower than in an array.

Lec 11 - Implementation of Priority Queue

- 1. Which of the following data structures is commonly used to implement a Priority Queue?
 - a) Linked List
 - b) Queue
 - c) Binary Heap
 - d) Stack

Solution: c) Binary Heap

- 2. Which of the following operations is commonly supported by a Priority Queue?
 - a) Enqueue
 - b) Dequeue
 - c) Insert
 - d) All of the above

Solution: d) All of the above

- 3. What is the time complexity of inserting an element into a binary heap?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) O(n log n)

Solution: b) O(log n)

- 4. What is the time complexity of deleting the highest-priority element from a binary heap?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) O(n log n)

Solution: b) O(log n)

- 5. Which of the following is an advantage of using a Fibonacci Heap to implement a Priority Queue?
 - a) Faster insert operation than a binary heap
 - b) Lower memory usage than a binary heap
 - c) Faster delete operation than a binary heap
 - d) All of the above

Solution: c) Faster delete operation than a binary heap

- 6. Which of the following algorithms makes use of a Priority Queue?
 - a) Dijkstra's shortest path algorithm
 - b) Binary search algorithm
 - c) Bubble sort algorithm
 - d) Linear search algorithm

Solution: a) Dijkstra's shortest path algorithm

- 7. Which of the following data structures is commonly used to implement a Priority Queue in C++?
 - a) std::queue
 - b) std::vector
 - c) std::list
 - d) std::priority_queue

Solution: d) std::priority_queue

8. Which of the following operations is not supported by a Priority Queue?

- a) Changing the priority of an element
- b) Inserting an element
- c) Removing an element with the lowest priority
- d) Removing an element with the highest priority

Solution: c) Removing an element with the lowest priority

9. Which of the following is an application of Priority Queues?

- a) Sorting large datasets
- b) Implementing a stack
- c) Implementing a queue
- d) Task scheduling in an operating system

Solution: d) Task scheduling in an operating system

10. Which of the following is a disadvantage of using a binary heap to implement a Priority Queue?

- a) Slower delete operation than a Fibonacci Heap
- b) Higher memory usage than a Fibonacci Heap
- c) Slower insert operation than a Fibonacci Heap
- d) All of the above

Solution: a) Slower delete operation than a Fibonacci Heap

Lec 12 - Operations on Binary Tree

- 1. What is the height of a binary tree with only one node?
 - a) 0
 - b) 1
 - c) 2
 - d) Undefined

Answer: a) 0

- 2. Which traversal method visits the nodes in the order left subtree, root, right subtree?
 - a) Pre-order traversal
 - b) In-order traversal
 - c) Post-order traversal
 - d) Level-order traversal

Answer: b) In-order traversal

- 3. Which traversal method visits the nodes in the order root, left subtree, right subtree?
 - a) Pre-order traversal
 - b) In-order traversal
 - c) Post-order traversal
 - d) Level-order traversal

Answer: a) Pre-order traversal

- 4. What is the time complexity of searching for a node in a Binary Tree?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) It depends on the implementation

Answer: c) O(n)

- 5. What is the maximum number of nodes at level k in a Binary Tree?
 - a) 2^k
 - b) k^2
 - c) k+1
 - d) None of the above

Answer: a) 2^k

- 6. Which of the following is a way to delete a node in a Binary Tree?
 - a) Deleting the node and its children
 - b) Replacing the node with its left child
 - c) Replacing the node with its right child
 - d) All of the above

Answer: d) All of the above

- 7. What is the time complexity of finding the height of a Binary Tree?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) It depends on the implementation

Answer: c) O(n)

- 8. What is the maximum number of nodes in a Binary Tree with height h?
 - a) 2^h
 - b) h^2
 - c) h+1
 - d) None of the above

Answer: a) 2^h - 1

- 9. Which of the following is a way to insert a node in a Binary Tree?
 - a) As the left child of a leaf node
 - b) As the right child of a leaf node
 - c) As the left child of a non-leaf node
 - d) All of the above

Answer: d) All of the above

- 10. Which of the following is an advantage of using a Binary Tree over a linked list?
 - a) Binary Tree can be searched faster than a linked list
 - b) Binary Tree can be sorted faster than a linked list
 - c) Binary Tree can store data in a hierarchical structure
 - d) All of the above

Answer: d) All of the above

Lec 13 - Cost of Search

1. What is the cost of search in computer science?

- a. Amount of time, resources, and computational power required to search for a specific item in a data structure
- b. The amount of data in a data structure
- c. The size of the data structure
- d. The type of data structure used

Answer: a.

- 2. Which of the following measures is a common measure of search cost?
- a. Time complexity
- b. Space complexity
- c. Worst-case analysis
- d. All of the above

Answer: d.

- 3. Which search algorithm is commonly used for searching a sorted array?
 - a. Binary search
 - b. Linear search
 - c. Depth-first search
 - d. Breadth-first search

Answer: a.

- 4. Which data structure is commonly used for implementing hash tables?
- a. Linked list
- b. Queue
- c. Stack
- d. Array

Answer: a.

- 5. What is the time complexity of binary search?
- a. O(1)
- b. O(log n)
- c. O(n)
- d. O(n log n)

Answer: b.

6. Which of the following is a disadvantage of linear search?

- a. It has a time complexity of O(1)
 b. It is easy to implement
 c. It has a time complexity of O(n)
 d. It is efficient for large datasets

 Answer: c.
 7. Which of the following is a disadvantage of binary search?

 a. It can only be used with sorted arrays
 b. It has a time complexity of O(n)
 c. It is difficult to implement
 d. It is not efficient for large datasets

 Answer: a.
 8. Which data attracture is commonly used for implementing binary search toos?
- 8. Which data structure is commonly used for implementing binary search trees?
- a. Array
- b. Linked list
- c. Queue
- d. Stack

Answer: b.

- 9. What is the time complexity of searching a hash table with a good hash function?
- a. O(1)
- b. O(log n)
- c. O(n)
- d. O(n log n)

Answer: a.

- 10. Which of the following measures is not a common measure of search cost?
 - a. Time efficiency
 - b. Space efficiency
 - c. Worst-case analysis
 - d. Best-case analysis

Answer: d.

Lec 14 - Recursive Calls

1. What is recursion?

- a. A process of dividing a problem into smaller subproblems
- b. A process of repeating a set of instructions
- c. A process of sorting data in ascending order
- d. None of the above

Answer: a

2. What is the base case in recursion?

- a. The case where the function calls itself
- b. The case where the function returns a value without calling itself
- c. The case where the function uses a loop instead of recursion
- d. None of the above

Answer: b

3. Which data structure is commonly used in recursion?

- a. Stack
- b. Queue
- c. Linked list
- d. Array

Answer: a

4. What is the maximum number of recursive calls that can be made?

- a. 100
- b. 1000
- c. It depends on the available memory
- d. There is no limit

Answer: c

5. Which of the following is a disadvantage of recursion?

- a. It is easy to understand and implement
- b. It may cause stack overflow errors
- c. It always results in better performance than iterative solutions
- d. None of the above

Answer: b

6. Which of the following algorithms uses recursion?

- a. Quick sort
- b. Merge sort
- c. Bubble sort
- d. Selection sort

Answer: a and b

7. Which of the following is true about recursive functions?

- a. They are always faster than iterative functions
- b. They require less memory than iterative functions
- c. They may be more readable and concise than iterative functions
- d. None of the above

Answer: c

8. What is tail recursion?

- a. A type of recursion where the recursive call is the last operation performed by the function
- b. A type of recursion where the function calls itself multiple times
- c. A type of recursion where the function uses a loop instead of recursion
- d. None of the above

Answer: a

9. What is the difference between direct and indirect recursion?

- a. Direct recursion occurs when a function calls itself, while indirect recursion occurs when two or more functions call each other.
- b. Direct recursion occurs when two or more functions call each other, while indirect recursion occurs when a function calls itself.
- c. Direct and indirect recursion are the same thing.
- d. None of the above

Answer: a

10. Which of the following is an example of a recursive data structure?

- a. Linked list
- b. Stack
- c. Queue
- d. Array

Answer: a and b

Lec 15 - Level-order Traversal of a Binary Tree

1. What is level-order traversal of a binary tree?

- A. Visiting the root node first
- B. Visiting the nodes level by level
- C. Visiting the left child first
- D. Visiting the right child first

Answer: B

2. Which data structure can be used to implement level-order traversal?

- A Stack
- B. Queue
- C. Linked list
- D. Binary search tree

Answer: B

3. What is the time complexity of level-order traversal?

- A. O(log n)
- B. O(n)
- C. O(n^2)
- D. O(2ⁿ)

Answer: B

4. In which order are the nodes visited in level-order traversal?

- A. Left to right, bottom to top
- B. Right to left, top to bottom
- C. Left to right, top to bottom
- D. Right to left, bottom to top

Answer: C

5. Which traversal technique can be used to print the nodes of a binary tree in level-order?

- A. In-order traversal
- B. Post-order traversal
- C. Pre-order traversal
- D. Level-order traversal

Answer: D

6. What is the space complexity of level-order traversal?

- A. O(1)
- B. O(n)
- C. O(n^2)
- D. O(2ⁿ)

Answer: B

7. Level-order traversal can be used to solve which type of problem?

- A. Finding the maximum depth of a binary tree
- B. Finding the minimum depth of a binary tree
- C. Finding the sum of all nodes in a binary tree
- D. Finding the lowest common ancestor of two nodes in a binary tree

Answer: A

8. Which of the following is an advantage of level-order traversal?

- A. It is faster than other traversal techniques
- B. It uses less memory than other traversal techniques
- C. It can be used to find the shortest path between two nodes
- D. It can be used to sort the nodes in a binary tree

Answer: C

9. What is the main disadvantage of level-order traversal?

- A. It is difficult to implement
- B. It requires more memory than other traversal techniques
- C. It is slower than other traversal techniques
- D. It does not work for binary trees with an odd number of nodes

Answer: B

10. What is the first node visited in level-order traversal?

- A. The root node
- B. The left child of the root node
- C. The right child of the root node
- D. It depends on the binary tree

Answer: A

Lec 16 - Deleting a node in BST

- 1. In a BST, which node is deleted when the node to be deleted has no children?
 - a) The root node
 - b) The node to be deleted
 - c) The parent of the node to be deleted
 - d) None of the above

Answer: b) The node to be deleted

- 2. When deleting a node with one child in a BST, which child of the deleted node replaces it?
 - a) The left child
 - b) The right child
 - c) It depends on the node's value
 - d) None of the above

Answer: c) It depends on the node's value

- 3. When deleting a node with two children in a BST, which node is used to replace the deleted node?
 - a) The left child of the deleted node
 - b) The right child of the deleted node
 - c) The smallest node in the right subtree of the deleted node
 - d) The largest node in the left subtree of the deleted node

Answer: c) The smallest node in the right subtree of the deleted node

- 4. Which traversal algorithm is commonly used to delete a node in a BST?
 - a) Inorder traversal
 - b) Preorder traversal
 - c) Postorder traversal
 - d) Level-order traversal

Answer: a) Inorder traversal

- 5. In a BST, what is the time complexity of deleting a node with one child?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) It depends on the height of the tree

Answer: b) O(log n)

- 6. What is the time complexity of deleting a node with two children in a BST?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) It depends on the height of the tree

Answer: d) It depends on the height of the tree

- 7. What happens when a leaf node is deleted in a BST?
 - a) The node is deleted and the tree is balanced
 - b) The node is deleted and the tree is left unbalanced
 - c) The tree becomes a binary tree
 - d) None of the above

Answer: a) The node is deleted and the tree is balanced

- 8. In a self-balancing BST, what type of rotation is performed when deleting a node with one child?
 - a) Left rotation
 - b) Right rotation
 - c) Double rotation
 - d) No rotation is performed

Answer: d) No rotation is performed

- 9. When deleting a node in a BST, what is the worst-case time complexity if the tree is unbalanced?
 - a) O(1)
 - b) O(log n)
 - c) O(n)
 - d) It depends on the size of the tree

Answer: c) O(n)

- 10. In a BST, what is the minimum number of children a node can have?
 - a) 0
 - b) 1
 - c) 2
 - d) There is no minimum number of children

Answer: a) 0

Lec 17 - Reference Variables

- 1. Which of the following data types is a reference type in Java?
 - a) int
 - b) double
 - c) String
 - d) char

Answer: c) String

- 2. What does a reference variable in Java hold?
 - a) The actual value of an object
 - b) The memory address of an object
 - c) The size of an object
 - d) The type of an object

Answer: b) The memory address of an object

- 3. When an object is assigned to a reference variable, what is stored in the variable?
 - a) A copy of the object
 - b) A reference to the object
 - c) The value of the object
 - d) None of the above

Answer: b) A reference to the object

- 4. What is the difference between a reference variable and a primitive variable?
 - a) A reference variable holds the actual value of an object, while a primitive variable holds a reference to an object.
 - b) A reference variable holds a reference to an object, while a primitive variable holds the actual value of a data type.
 - c) A reference variable holds a value of an object, while a primitive variable holds a reference to an object.
 - d) There is no difference between the two.

Answer: b) A reference variable holds a reference to an object, while a primitive variable holds the actual value of a data type.

- 5. In Java, can a reference variable be null?
 - a) Yes
 - b) No

Answer: a) Yes

- 6. What happens when a reference variable is assigned to another reference variable?
 - a) Both variables hold a reference to the same object
 - b) Both variables hold a copy of the same object
 - c) The original variable is deleted
 - d) None of the above

Answer: a) Both variables hold a reference to the same object

- 7. What is the syntax for creating a reference variable in Java?
 - a) int x = 5;
 - b) double y = 3.14;
 - c) String s = "Hello";
 - d) char c = 'a';

Answer: c) String s = "Hello";

- 8. What happens when a reference variable is passed as a parameter to a method in Java?
 - a) The method receives a copy of the object
 - b) The method receives a reference to the object
 - c) The method receives the actual value of the object
 - d) None of the above

Answer: b) The method receives a reference to the object

- 9. What is the default value of a reference variable in Java?
 - a) null
 - b) 0
 - c) false
 - d) None of the above

Answer: a) null

- 10. Can a reference variable be used to access static methods in Java?
 - a) Yes
 - b) No

Answer: a) Yes

Lec 18 - Reference Variables

1. What is a reference variable in Java?

- a. A variable that holds the actual value of a data type.
- b. A variable that holds the memory address of an object.
- c. A variable that holds both the value of a data type and the memory address of an object.

Answer: b

2. What is the default value of a reference variable in Java?

- a. 0
- b. false
- c. null

Answer: c

3. Can a reference variable be reassigned to a different object in Java?

- a. Yes
- b. No
- c. It depends on the data type of the reference variable.

Answer: a

4. How does Java handle passing a reference variable as a parameter to a method?

- a. It passes the actual value of the reference variable.
- b. It passes the memory address of the object held by the reference variable.
- c. It does not allow passing reference variables as parameters to methods.

Answer: b

5. Can a reference variable be used to access static methods in Java?

- a. Yes
- b. No
- c. It depends on the access modifier of the static method.

Answer: a

6. How does Java handle garbage collection for objects referenced by reference variables?

- a. It automatically frees up memory allocated to objects that are no longer being referenced.
- b. It requires manual intervention to free up memory allocated to objects.
- c. It does not perform garbage collection for objects referenced by reference variables.

Answer: a

7. What is the difference between an instance variable and a reference variable in Java?

- a. An instance variable holds the memory address of an object, while a reference variable is a variable declared in a class.
- b. An instance variable is a variable declared in a method, while a reference variable is a variable declared in a class.
- c. An instance variable is a variable declared in a class, while a reference variable is a variable declared in a method or block that holds a reference to an object.

Answer: c

8. Can a reference variable be used to access private members of a class in Java?

- a. Yes
- b. No
- c. It depends on the access modifier of the private member.

Answer: b

9. How can we check if a reference variable is null in Java?

- a. By using the null keyword.
- b. By using the equals() method.
- c. By using the == operator.

Answer: c

10. Can a reference variable be assigned to a primitive value in Java?

- a. Yes
- b. No
- c. It depends on the data type of the reference variable.

Answer: b

Lec 19 - Usage of const keyword

- 1. What does the const keyword do in C++?
 - a) Declares a variable that cannot be modified
 - b) Declares a variable that can only be modified in specific contexts
 - c) Declares a variable that is used for debugging purposes

Answer: a

- 2. Which of the following is a benefit of using const variables?
 - a) Improved program performance
 - b) Increased memory usage
 - c) Reduced bugs and improved program stability

Answer: c

- 3. Can const variables be modified after they are initialized?
 - a) Yes
 - b) No

Answer: b

- 4. What happens if you try to modify a const variable in C++?
 - a) The program crashes
 - b) The compiler generates an error
 - c) The modification is allowed

Answer: b

- 5. Which of the following types of variables is typically declared as const in C++?
 - a) Variables that are used for input/output
 - b) Global variables
 - c) Variables that are used as constants in the program

Answer: c

- 6. Is the const keyword required when passing a variable by reference in C++?
 - a) Yes
 - b) No

Answer: a

- 7. Can a member function of a C++ class be declared as const?
 - a) Yes
 - b) No

Answer: a

- 8. Which of the following is an example of a const pointer in C++?
 - a) int* const ptr;
 - b) const int* ptr;
 - c) const int* const ptr;

Answer: a

- 9. Can a const variable be initialized with a value at runtime in C++?
 - a) Yes
 - b) No

Answer: b

10.	Is the const keyword used in other programming languages besides C++?
	a) Yes
	b) No
	Answer: a

Lec 20 - AVL Tree

- 1. What is AVL Tree?
 - a) Binary Tree
 - b) Self-balancing Binary Search Tree
 - c) Hash Tree
 - d) None of the above

Answer: b) Self-balancing Binary Search Tree

- 2. In AVL Tree, what is the maximum difference between the height of the left and right subtrees?
 - a) 1
 - b) 2
 - c) 3
 - d) 4

Answer: a) 1

- 3. What is the time complexity of search operation in AVL Tree?
 - a) O(log n)
 - b) O(n)
 - c) O(n log n)
 - d) O(1)

Answer: a) O(log n)

- 4. In AVL Tree, what operation is performed to balance the tree?
 - a) Rotation
 - b) Inversion
 - c) Deletion
 - d) None of the above

Answer: a) Rotation

- 5. What is the worst-case time complexity of insertion operation in AVL Tree?
 - a) O(log n)
 - b) O(n)
 - c) O(n log n)
 - d) O(1)

Answer: a) O(log n)

- 6. What is the height of an AVL Tree with n nodes in the worst-case scenario?
 - a) log(n)
 - b) log(n) + 1
 - c) 2log(n)
 - d) $2\log(n) + 1$

Answer: b) log(n) + 1

7. Which of the following statements is true about AVL Tree?

- a) AVL Tree is a balanced binary search tree
- b) AVL Tree is an unbalanced binary search tree
- c) AVL Tree is a type of heap data structure
- d) AVL Tree is a type of graph data structure

Answer: a) AVL Tree is a balanced binary search tree

8. What is the time complexity of deletion operation in AVL Tree?

- a) O(log n)
- b) O(n)
- c) O(n log n)
- d) O(1)

Answer: a) O(log n)

9. Which of the following is not a balancing rule in AVL Tree?

- a) Right-Right (RR)
- b) Left-Right (LR)
- c) Left-Left (LL)
- d) Right-Left (RL)

Answer: d) Right-Left (RL)

10. Can AVL Tree have duplicate keys?

- a) Yes
- b) No

Answer: b) No

Answer: b) 4

L	ec 21 - AVL Tree Building Example							
1.	What is the first node added to the AVL Tree in the building example? a) 1 b) 2 c) 5 d) 8							
Ar	Answer: c) 5							
2.	What is the second node added to the AVL Tree in the building example? a) 1 b) 2 c) 3 d) 8							
Ar	nswer: b) 2							
3.	How many rotations are performed to maintain balance after inserting node 1 in the AVL Tree? a) 0 b) 1 c) 2 d) 3							
Answer: b) 1								
4.	What is the height of the AVL Tree after inserting node 3? a) 1 b) 2 c) 3 d) 4							
Ar	nswer: b) 2							
5.	Which rotation is performed after inserting node 6 to maintain balance in the AVL Tree? a) Left rotation b) Right rotation c) Left-right rotation d) Right-left rotation							
Ar	nswer: a) Left rotation							
6.	What is the height of the AVL Tree after inserting node 9? a) 3 b) 4 c) 5 d) 6							

What is the root node of the AVL Tree after inserting all the nodes? a) 1 b) 2 c) 5 d) 8 nswer: c) 5						
Which is the last node added to the AVL Tree in the building example? a) 6 b) 8 c) 9 d) 3						
Answer: c) 9						
What is the maximum height of an AVL Tree with 7 nodes? a) 2 b) 3 c) 4 d) 5						
Answer: b) 3						
How many rotations are performed in total to maintain balance while building the AVL Tree in the example? a) 2 b) 3 c) 4 d) 5						
nswer: c) 4						

Lec 22 - Cases of rotations

1. Which of the following is not a type of rotation in a binary search tree?

- A. Left rotation
- B. Right rotation
- C. Upward rotation
- D. Double rotation

Answer: C

2. When is a single left rotation used in a binary search tree?

- A. When the imbalance occurs in the immediate left child
- B. When the imbalance occurs in the immediate right child
- C. When the imbalance occurs in the grandchild of the left child
- D. When the imbalance occurs in the grandchild of the right child

Answer: A

3. Which of the following is a case of double rotation in a binary search tree?

- A. Left-Left case
- B. Left-Right case
- C. Right-Left case
- D. Right-Right case

Answer: B

4. In a left-right double rotation, what is the first step performed?

- A. A single left rotation on the right child
- B. A single right rotation on the left child
- C. A double right rotation on the left child
- D. A double left rotation on the right child

Answer: B

5. Which of the following is not a benefit of rotations in a binary search tree?

- A. Maintaining balance
- B. Ensuring efficient search operations
- C. Reducing the height of the tree
- D. Increasing the height of the tree

Answer: D

6. What is the maximum number of rotations required to balance a node in a binary search tree?

- A. 1
- B. 2
- C. 3
- D. 4

Answer: B

7. When is a single right rotation used in a binary search tree?

- A. When the imbalance occurs in the immediate left child
- B. When the imbalance occurs in the immediate right child
- C. When the imbalance occurs in the grandchild of the left child
- D. When the imbalance occurs in the grandchild of the right child

Answer: B

8. Which of the following is a case of single rotation in a binary search tree?

- A. Left-Left case
- B. Left-Right case
- C. Right-Left case
- D. Right-Right case

Answer: A

9. In a left-left single rotation, what is the new root of the subtree?

- A. The left child of the original root
- B. The right child of the original root
- C. The parent of the original root
- D. The original root itself

Answer: A

10. Which of the following is a disadvantage of using rotations in a binary search tree?

- A. Increased tree height
- B. Reduced search efficiency
- C. Increased complexity
- D. None of the above

Answer: D