CUET-UG Computer Science Sample Paper - 18

Duration: 1 Hour **Maximum Marks: 250**

Instructions

- This paper contains a total of 50 Multiple Choice Questions.
- Each correct answer carries **+5 marks**.
- Each incorrect answer carries **-1 mark**.
- No negative marking for unattempted questions.

Q1. Which of the following SQL queries will return the first three characters of the string 'COMPUTER' in uppercase?

(A) SELECT UPPER(LEFT('COMPUTER', 3));

(B) SELECT UPPER(SUBSTRING('COMPUTER', 1, 3));

(C) SELECT MID(UPPER('COMPUTER'), 1, 3);

(D) All of the above

Q2. Consider the SQL statement: `SELECT ROUND(157.48, -1)`. The result of this query will be:

(A) 157.5

(B) 160

(C) 150

(D) 157

Q3. Which SQL function is used to return the name of the month for a given date value?

(A) MONTHNAME()

(B) DAYNAME()

(C) MONTH()

(D) STR_MONTH()

Q4. What will be the output of `SELECT INSTR('DATABASE MANAGEMENT', 'A');`?

(A) 1

(B) 2

(C) 4

(D) 3

Q5. To remove only the trailing spaces from a string " Hello World ", which SQL function should be used?

(A) LTRIM()

(B) RTRIM()

(C) TRIM()

(D) REMOVE()

Q6. The SQL function `POW(X, Y)` returns:

(A) $X \times Y$

(B) X^Y

(C) Y^X

(D) $X \div Y$

Q7. Which of the following will result in '2026-05-07' if the current date is May 7, 2026?

(A) SELECT NOW();

(B) SELECT DATE(NOW());

(C) SELECT CURDATE();

(D) Both (B) and (C)

Q8. What is the output of `SELECT MOD(13, 5) + TRUNCATE(15.79, 1);`?

(A) 18.7

(B) 18.3

(C) 3 + 15.7

(D) 18.0

Q9. In Relational Algebra, which operator is used to select a subset of columns from a table?

(A) Selection (σ)

(B) Projection (π)

(C) Join ($\bowtie$)

(D) Union ($\cup$)

Q10. A _____ is a minimal set of attributes that uniquely identifies a tuple in a relation.

(A) Primary Key

(B) Foreign Key

(C) Candidate Key

(D) Composite Key

Q11. The property that ensures a database remains in a consistent state even after a transaction fails is known as:

(A) Atomicity

(B) Consistency

(C) Isolation

(D) Durability

Q12. Which of the following constraints prevents the entry of duplicate values in a column but allows a NULL value?

(A) PRIMARY KEY

(B) UNIQUE

(C) NOT NULL

(D) CHECK

Q13. Which network topology requires a central controller or hub to which all nodes are connected?

(A) Mesh

(B) Ring

(C) Star

(D) Bus

Q14. A MAC address is a _____ bit hardware address assigned to a Network Interface Card.

(A) 32

(B) 64

(C) 48

(D) 128

Q15. Which device operates at the Physical layer of the OSI model and regenerates signals to extend network distance?

(A) Router

(B) Bridge

(C) Switch

(D) Repeater

Q16. If the elements 'A', 'B', 'C' are pushed onto a stack in that order, what will be the order of elements when they are popped?

(A) A, B, C

(B) C, B, A

(C) B, A, C

(D) A, C, B

Q17. The postfix expression for the infix expression $(A + B) \times (C - D)$ is:

(A) $AB + CD - \times$

(B) $AB + CD \times -$

(C) $+AB - CD\times$

(D) $ABCD + -\times$

Q18. A queue follows which of the following principles?

(A) LIFO

(B) LILO

(C) FIFO

(D) Random Access

Q19. Evaluate the following postfix expression: $10, 2, \times, 5, +, 3, -$

(A) 22

(B) 25

(C) 17

(D) 28

Q20. In a Python list `L = [10, 20, 30, 40, 50]`, what is the result of `L.pop(2)`?

(A) 20

(B) 30

(C) [10, 20, 40, 50]

(D) 40

Q21. To implement a stack using a Python list, which functions are most commonly used for 'Push' and 'Pop'?

(A) push() and pop()

(B) insert() and delete()

(C) append() and pop()

(D) add() and remove()

Q22. What is the time complexity of accessing an element in a Python list by its index?

(A) $O(1)$

(B) $O(n)$

(C) $O(\log n)$

(D) $O(n^2)$

Q23. Which data structure is best suited for reversing a string?

(A) Queue

(B) Stack

(C) Linked List

(D) Tree

Q24. If a circular queue has a size of 5 and the Front is at index 3 and Rear is at index 2, the queue is:

(A) Empty

(B) Full

(C) Overflow

(D) Underflow

Q25. In a Stack, if the 'Top' pointer is at -1, it indicates:

(A) Overflow

(B) Underflow

(C) The stack has one element

(D) The stack is full

Q26. Which operation in a queue is used to insert a new element?

(A) Push

(B) Dequeue

(C) Enqueue

(D) Pop

Q27. What is the maximum number of comparisons needed to find an element in an array of 128 elements using Binary Search?

(A) 128

(B) 64

(C) 7

(D) 8

Q28. Which sorting algorithm repeatedly finds the minimum element from the unsorted part and puts it at the beginning?

(A) Bubble Sort

(B) Insertion Sort

(C) Selection Sort

(D) Quick Sort

Q29. In Bubble Sort, after the first pass through an array of n elements, which element is guaranteed to be in its correct position?

(A) The first element

(B) The last element

(C) The middle element

(D) None

Q30. The best-case time complexity of Linear Search is:

(A) $O(1)$

(B) $O(n)$

(C) $O(\log n)$

(D) $O(n^2)$

Q31. Which sorting algorithm is often compared to the way one sorts a deck of cards?

(A) Selection Sort

(B) Bubble Sort

(C) Insertion Sort

(D) Merge Sort

Q32. Binary Search requires the data to be in _____ order.

(A) Random

(B) Sorted

(C) Reverse

(D) Any

Q33. Which of the following is an unstable sorting algorithm?

(A) Bubble Sort

(B) Insertion Sort

(C) Selection Sort

(D) All are stable

Q34. What is the average case time complexity of Selection Sort?

(A) $O(n)$

(B) $O(n^2)$

(C) $O(n \log n)$

(D) $O(\log n)$

Q35. How many passes are required to sort an array of 6 elements using Bubble Sort?

(A) 6

(B) 5

(C) 36

(D) 4

Q36. Which block in Python is used to catch and handle exceptions?

(A) catch

(B) except

(C) throw

(D) error

(E) try

Q37. The `finally` block in Python is executed:

(A) Only if an exception occurs

(B) Only if no exception occurs

(C) Always, regardless of an exception

(D) Only if the try block is empty

Q38. To open a file for reading in binary mode, which mode string should be used?

(A) 'r'

(B) 'rb'

(C) 'w'

(D) 'wb'

Q39. Which method is used to write a list of strings into a text file?

(A) write()

(B) writelines()

(C) putlines()

(D) dump()

Q40. When opening a file using `with open('data.txt', 'r') as f:`, what is the advantage?

(A) It runs faster

(B) It automatically closes the file

(C) It encrypts the file

(D) It creates the file if it doesn't exist

Q41. Which module is required to perform input/output operations on binary files (pickling)?

(A) os

(B) math

(C) pickle

(D) sys

Q42. What does the `seek(5, 0)` method do in file handling?

(A) Moves the pointer 5 bytes from the current position

(B) Moves the pointer 5 bytes from the end of the file

(C) Moves the pointer 5 bytes from the beginning of the file

(D) Deletes the first 5 bytes

Q43. Which error is raised when a file to be opened for reading does not exist?

(A) IOError

(B) FileNotFoundError

(C) ValueError

(D) ImportError

Q44. Which protocol is primarily used for transferring files between a client and a server on a network?

(A) HTTP

(B) SMTP

(C) FTP

(D) POP3

Q45. In which type of switching is a dedicated physical path established between two nodes before data transfer?

(A) Packet Switching

(B) Message Switching

(C) Circuit Switching

(D) Frequency Switching

Q46. A _____ is a malicious program that attaches itself to another program and replicates when the host program is executed.

(A) Worm

(B) Virus

(C) Trojan Horse

(D) Spyware

Q47. Which security threat involves a person pretending to be someone else to gain unauthorized access to data?

(A) Phishing

(B) Eavesdropping

(C) Spoofing

(D) Denial of Service

Q48. What is the primary function of a Firewall?

(A) To speed up internet connection

(B) To monitor and control incoming/outgoing network traffic

(C) To store backups of data

(D) To provide a wireless signal

Q49. Which protocol is used to secure data over the web by providing encryption (denoted by a padlock icon in browsers)?

(A) HTTP

(B) HTTPS

(C) IP

(D) TCP

Q50. The practice of sending fraudulent emails that appear to be from reputable companies to induce individuals to reveal personal information is:

(A) Hacking

(B) Phishing

(C) Spamming

(D) Cracking

Detailed Solutions

Q1.

Solution

Concept: This question explores the application of SQL string manipulation functions, specifically focusing on how to extract and transform substrings. In SQL (specifically MySQL, which is the standard for CUET), multiple functions can often achieve the same result. The functions involved here are `UPPER()`, `LEFT()`, `SUBSTRING()`, and `MID()`.

Solution: 1. **Analysis of Option (A):** The query is `SELECT UPPER(LEFT('COMPUTER', 3));`.

- First, the inner function `LEFT('COMPUTER', 3)` is executed. This function extracts the first three characters from the left side of the string, resulting in 'COM'.
- Next, the outer function `UPPER('COM')` is applied. Since 'COM' is already in uppercase, it remains 'COM'. If the input were lowercase, it would have been converted. This correctly returns the first three characters in uppercase.

2. **Analysis of Option (B):** The query is `SELECT UPPER(SUBSTRING('COMPUTER', 1, 3));`.

- The `SUBSTRING(string, start, length)` function is used here. In SQL, the starting index is 1. So, `SUBSTRING('COMPUTER', 1, 3)` starts at the first character ('C') and takes a length of 3 characters, resulting in 'COM'.
- The `UPPER()` function then ensures the output is capitalized. This also correctly returns 'COM'.

3. **Analysis of Option (C):** The query is `SELECT MID(UPPER('COMPUTER'), 1, 3);`.

- Here, the `UPPER()` function is applied first to the whole string 'COMPUTER'.
- Then, the `MID()` function (which is a synonym for `SUBSTRING()` in MySQL) extracts 3 characters starting from position 1. This results in 'COM'.

4. **Conclusion:** All three queries, despite their slightly different syntax and order of operations, logically result in the same output: the first three characters of the word 'COMPUTER' in uppercase format.

Final Answer: All of the above

Answer: (D)

Q2.

Solution

Concept: The `ROUND()` function in SQL is a mathematical function used to round a numeric value to a specific number of decimal places. A crucial feature of this function is its behavior when the second argument (the scale) is a negative integer. While positive values round to the right of the decimal point, negative values round to the left of the decimal point (the whole number parts).

Solution: 1. **Identifying the Parameters:** The input value is 157.48 and the rounding factor is -1.

2. **Understanding Negative Rounding:**
- A rounding factor of 0 would round to the nearest whole number (integer).
- A rounding factor of -1 tells the database to round to the nearest 10 (the tens place).
- A rounding factor of -2 would round to the nearest 100 (the hundreds place).

3. **Applying the Logic to 157.48:**
- We look at the digit to the right of the place we are rounding to. Since we are rounding to the tens place (where the digit is 5), we look at the units place (the digit 7).
- Standard rounding rules apply: if the digit is 5 or greater, we round up. Since $7 \geq 5$, we increment the tens digit.
- The digit 5 in the tens place becomes 6, and the units place (and everything to the right of the decimal) becomes 0.

4. **Calculating the Result:**
- Rounding 157.48 to the nearest ten results in 160.
- If the input had been 154.48, the result would have been 150 because 4 is less than 5.
- Because the units digit is 7, the value is closer to 160 than to 150.

Final Answer: 160

Answer: (B)

Q3.

Solution

Concept: Working with Temporal Data (Date and Time) is a core component of SQL. Database systems like MySQL provide specialized functions to extract specific components from a `DATE` or `DATETIME` value. Understanding the difference between numerical extraction and name extraction is vital for generating user-friendly reports.

Solution: 1. **Requirement Analysis:** The question asks for a function that returns the ”name” of the month (e.g., ’January’, ’February’) rather than its numerical index (e.g., 1, 2).

2. **Evaluating Function Options:**

- `MONTH(date)`: This function extracts the numerical month from a date. For a date like ’2026-05-07’, it would return the integer 5. This does not meet the ”name” requirement.
- `MONTHNAME(date)`: This is the specific function built to return the full name of the month as a string. For ’2026-05-07’, it returns ’May’. This aligns perfectly with the question.
- `DAYNAME(date)`: This function returns the name of the day of the week (e.g., ’Thursday’). While similar in naming convention, it extracts the wrong part of the date.
- `STR_MONTH()`: This is not a standard built-in SQL function in MySQL or PostgreSQL; it is a distractor.

3. **Practical Application:** In a real-world scenario, if a developer wants to create a header for a monthly report, they would use `SELECT MONTHNAME(CURDATE());` to display the current month in a readable format.

4. **Summary:** The `MONTHNAME()` function is the correct choice because it is designed specifically for string representation of the month component of a date.

Final Answer: MONTHNAME()

Answer: (A)

Q4.

Solution

Concept: The `INSTR()` function (In-String) is used to search for the first occurrence of a substring within a larger string. It is a vital tool for text analysis in databases. A key distinction in SQL compared to many programming languages like Python or C++ is that string indexing is 1-based, meaning the very first character of the string is at position 1, not 0.

Solution: 1. **Function Breakdown:** The syntax is `INSTR(string, target_substring)`. The function scans the `string` from left to right and returns the integer position of the first character of the `target_substring`.

2. **Step-by-Step Character Scan:**
- Let's map the string 'DATABASE MANAGEMENT':
- Index 1: 'D'
- Index 2: 'A' ← First match found!
- Index 3: 'T'
- Index 4: 'A'
- Index 5: 'B'
- (and so on...)

3. **Handling Multiple Occurrences:** The string 'DATABASE MANAGEMENT' actually contains the character 'A' four times (at positions 2, 4, 11, and 14). However, the logic of the `INSTR()` function is to return the position of the *first* occurrence and then terminate the search.

4. **Result Determination:** Since the first 'A' is found at the second character of the string, the function returns the value 2. If the substring was not found at all, the function would return 0. In this case, the result is clearly 2.

Final Answer: 2

Answer: (B)

Q5.

Solution

Concept: Data cleaning often involves removing extraneous spaces that might be included during data entry. SQL provides ”Trim” functions to handle different parts of a string. These are essential for ensuring that string comparisons (like in a `WHERE` clause) work correctly, as ’Hello’ is not equal to ’Hello ’.

Solution: 1. **Understanding the Variants of Trim:**

- `TRIM()`: This function removes all leading spaces (at the start) and all trailing spaces (at the end). It effectively ”cleans” both sides.
- `LTRIM()`: Short for ”Left Trim,” this function only removes spaces from the beginning of the string (the left side).
- `RTRIM()`: Short for ”Right Trim,” this function only removes spaces from the end of the string (the right side).

2. **Applying to the Scenario:** The question specifically asks to remove ”only the trailing spaces” from the string `" Hello World "`.

- The input has 3 leading spaces and 3 trailing spaces.
- If we use `RTRIM()`, the 3 leading spaces will remain, but the 3 trailing spaces will be deleted.
- The resulting string would be `" Hello World"`.

3. **Evaluating Distractors:**

- `LTRIM()` would leave the trailing spaces, which is the opposite of the requirement.
- `TRIM()` would remove both, but the question asks to target ”only” the trailing ones.
- `REMOVE()` is not a valid SQL function for this operation.

4. **Conclusion:** For targeting the end of the string specifically, `RTRIM()` is the correct and most efficient choice.

Final Answer: RTRIM()

Answer: (B)

Q6.

Solution

Concept: The `POW(X, Y)` function, also written as `POWER(X, Y)`, is a built-in mathematical function used in SQL to perform exponentiation. In computational mathematics within a database, understanding the order of parameters is vital because the operation is non-commutative (i.e., X^Y is not the same as Y^X unless $X = Y$). This function is widely used in scientific calculations and financial modeling within SQL queries.

Solution: 1. **Parameter Identification:** In the expression `POW(X, Y)`, X is the base and Y is the exponent.

2. **Mathematical Translation:** The function computes the value of X multiplied by itself Y times. In mathematical notation, this is represented as X^Y.

3. **Verification with Examples:** - If we execute `SELECT POW(5, 2);`, the engine calculates 5×5, which equals 25. - If we execute `SELECT POW(2, 5);`, the engine calculates $2 \times 2 \times 2 \times 2 \times 2$, which equals 32.

4. **Logic Check:** - Option (A) $X \times Y$ is simple multiplication ($5 \times 2 = 10$). - Option (B) X^Y matches our derivation. - Option (C) Y^X would be the reverse (2^5). - Option (D) $X \div Y$ is division.

5. **Conclusion:** The logic of the `POW` function is strictly to raise the first argument to the power of the second argument. This ensures that the base-exponent relationship is maintained consistently across SQL dialects like MySQL and PostgreSQL.

Final Answer: X^Y

Answer: (B)

Q7.

Solution

Concept: Date and Time functions are essential for managing temporal data in databases. This question focuses on the distinction between retrieving a full timestamp (date and time) and retrieving a formatted or specific date value. In SQL, especially MySQL, there are multiple paths to achieve a 'YYYY-MM-DD' output, and understanding how these functions nest or stand alone is key for CUET-UG Computer Science.

Solution: 1. **Analyzing NOW():** The `NOW()` function returns the current system date and time. If today is May 7, 2026, `NOW()` returns '2026-05-07 09:55:00'. This does not match the required string '2026-05-07' exactly because it contains the time component.

2. **Analyzing DATE(NOW()):** The `DATE()` function is an extractor. It takes a `DATETIME` expression as an argument and returns only the date part. Therefore, `DATE('2026-05-07 09:55:00')` results in '2026-05-07'. This is a valid method.

3. **Analyzing CURDATE():** The `CURDATE()` function is specifically designed to return the current date in 'YYYY-MM-DD' format without any time information. It is functionally equivalent to `CURRENT_DATE`. This also results in '2026-05-07'.

4. **Step-by-Step Validation:** - Option (B) works by stripping the time from the current timestamp. - Option (C) works by calling the native current date function. - Since both (B) and (C) produce the desired output, the correct choice must account for both.

5. **Final Deduction:** In professional SQL practice, `CURDATE()` is more common for readability, but `DATE(NOW())` is logically sound. Both meet the criteria.

Final Answer: Both (B) and (C)

Answer: (D)

Q8.

Solution

Concept: This question tests the user's ability to evaluate complex expressions involving multiple SQL mathematical functions. It requires understanding the `MOD()` function (Modulo) and the `TRUNCATE()` function. A common point of failure is confusing `TRUNCATE()` with `ROUND()`, as they behave differently regarding the fractional part of a number.

Solution: 1. **Solving Part 1: MOD(13, 5):** - The modulo function returns the remainder of a division. - $13 \div 5 = 2$ with a remainder of 3 (since $5 \times 2 = 10$, and $13 - 10 = 3$). - Therefore, `MOD(13, 5) = 3`.

2. **Solving Part 2: TRUNCATE(15.79, 1):** - The `TRUNCATE` function cuts off a number at a specific decimal place without rounding up. - The value is 15.79 and the scale is 1. This means we keep one digit after the decimal point and discard the rest. - Unlike `ROUND(15.79, 1)`, which would give 15.8 because $9 \geq 5$, `TRUNCATE` simply removes the 9. - Therefore, `TRUNCATE(15.79, 1) = 15.7`.

3. **Final Calculation:** - We now perform the addition as requested: $3 + 15.7$. - $3.0 + 15.7 = 18.7$.

4. **Reasoning Check:** If we had used `ROUND`, the answer would have been 18.8. If we had used integer division, the answer would have been different. The specificity of "Truncate" is the key to this question.

Final Answer: 18.7

Answer: (A)

Q9.

Solution

Concept: Relational Algebra provides the formal logic for the operations we perform in SQL. There are two primary unary operations: Selection (σ) and Projection (π). Selection filters the data horizontally (rows), whereas Projection filters the data vertically (columns). Understanding this distinction is fundamental to database theory.

Solution: 1. **Identifying the Target:** The question asks for an operator that selects a "subset of columns." This means we are discarding some attributes and keeping others.

2. **Defining Selection (σ):** Selection is used to choose tuples (rows) that satisfy a certain predicate. In SQL, this is equivalent to the `WHERE` clause. It does not reduce the number of columns.

3. **Defining Projection (π):** Projection is used to choose specific attributes (columns). In SQL, this is represented by the `SELECT` clause followed by column names. For example, $\pi_{Name,Age}(Students)$ will only show the Name and Age columns.

4. **Reviewing other Operators:** - **Join ($\bowtie$):** Combines rows from two or more tables based on a related column. - **Union ($\cup$):** Combines the result-set of two or more queries.

5. **Conclusion:** Since the operation specifically describes choosing columns, the only theoretically correct answer in Relational Algebra is the Projection operator.

Final Answer: Projection (π)

Answer: (B)

Q10.

Solution

Concept: Database keys are constraints that identify uniqueness within a relation. There is a hierarchy of keys: Super Keys, Candidate Keys, and Primary Keys. A Candidate Key is a specific type of key that must satisfy two properties: it must uniquely identify the row, and it must be minimal. ”Minimal” means that no proper subset of that key can uniquely identify the row.

Solution: 1. **Attribute Uniqueness:** In any table, we need a way to distinguish one record from another. A set of attributes that does this is a ”Super Key.”

2. **The Minimality Constraint:** If a Super Key has no redundant attributes (i.e., you cannot remove any attribute from it without losing the uniqueness property), it is upgraded to a ”Candidate Key.”

3. **Step-by-Step Example:** - Suppose we have a table with $\{Student\_ID, Email, Name\}$. - $\{Student\_ID, Name\}$ is a Super Key but not a Candidate Key (because $Name$ is redundant). - $\{Student\_ID\}$ is a Candidate Key because it is minimal. - $\{Email\}$ is also a Candidate Key.

4. **Selection of Primary Key:** From the available pool of Candidate Keys, the DBA chooses one to be the Primary Key. However, the definition of the ”minimal set” itself refers broadly to the Candidate Key.

5. **Conclusion:** Because the question emphasizes the ”minimal set of attributes” that uniquely identify a tuple, the term ”Candidate Key” is the most accurate formal definition.

Final Answer: Candidate Key

Answer: (C)

Q11.

Solution

Concept: Database transactions are governed by the ACID properties: Atomicity, Consistency, Isolation, and Durability. This question focuses specifically on the ”Consistency” and ”Atomicity” aspects of transaction management. In a database environment, a transaction is a logical unit of work that must either be completed in its entirety or not at all to ensure the database remains in a valid state.

Solution: 1. **Analyzing Atomicity:** This property ensures that all operations within a transaction are completed. If any part of the transaction fails, the entire transaction is rolled back, and the database is left unchanged. This is the ”all or nothing” rule.

2. **Analyzing Consistency:** Consistency ensures that a transaction takes the database from one valid state to another valid state, maintaining all predefined rules (constraints, cascades, triggers). If a transaction fails, the system must revert to its last consistent state.

3. **Analyzing Isolation:** This ensures that concurrent execution of transactions leaves the database in the same state as if they were executed sequentially.

4. **Analyzing Durability:** This guarantees that once a transaction has been committed, it will remain so, even in the event of a power loss or system crash.

5. **Detailed Reasoning:** The question specifies the property that ”ensures a database remains in a consistent state even after a transaction fails.” While Atomicity handles the rollback mechanism, the overall requirement to maintain a valid, rule-abiding state is defined by the Consistency property. When a transaction violates integrity constraints, it is aborted to preserve this consistency.

Final Answer: Consistency

Answer: (B)

Q12.

Solution

Concept: SQL constraints are rules applied to table columns to limit the type of data that can be entered. This ensures the accuracy and reliability of the data. Two commonly confused constraints are `PRIMARY KEY` and `UNIQUE`. Both enforce uniqueness, but they handle NULL values differently, which is a critical distinction in database design.

Solution: 1. **Evaluating PRIMARY KEY:** A `PRIMARY KEY` constraint uniquely identifies each record in a table. By definition, a primary key column cannot contain `NULL` values and must be unique. A table can have only one primary key.

2. **Evaluating UNIQUE:** The `UNIQUE` constraint ensures that all values in a column are different from each other. However, unlike the primary key, it allows for the presence of `NULL` values (depending on the database system, multiple NULLs might even be allowed as NULL is treated as an unknown value).

3. **Evaluating NOT NULL:** This constraint simply ensures that a column cannot have a `NULL` value, but it does not enforce uniqueness on the values (e.g., multiple people can have the same 'City').

4. **Evaluating CHECK:** This constraint ensures that all values in a column satisfy a specific condition (e.g., `Age >= 18`).

5. **Conclusion:** Since the question explicitly mentions a constraint that "prevents duplicate values but allows a NULL value," the `UNIQUE` constraint is the only one that fits both criteria perfectly.

Final Answer: UNIQUE

Answer: (B)

Q13.

Solution

Concept: Network topology refers to the physical or logical arrangement of nodes (computers, printers, etc.) in a network. Different topologies offer various levels of redundancy, cost, and ease of installation. The Star topology is one of the most common configurations in modern Local Area Networks (LANs), utilizing a central connection point.

Solution: 1. **Analyzing Mesh Topology:** In a full mesh, every node is connected to every other node. There is no central hub; instead, there is massive redundancy.

2. **Analyzing Ring Topology:** Nodes are connected in a closed loop. Each node is connected to exactly two neighbors, and data travels in one direction.

3. **Analyzing Bus Topology:** All nodes share a single communication line (the backbone). Data is broadcasted to all nodes, and terminators are used at the ends.

4. **Analyzing Star Topology:** In this configuration, every peripheral node is connected to a central node, usually a Hub, Switch, or Router. All data transmission goes through this central controller, which acts as a signal repeater or director.

5. **Step-by-Step Logic:** - Does it have a central hub? Yes (Star). - Is it the most common for Ethernet LANs? Yes. - If the hub fails, does the whole network go down? Yes. 6. **Conclusion:** The description of a ”central controller or hub” is the defining characteristic of the Star topology.

Final Answer: Star

Answer: (C)

Q14.

Solution

Concept: A Media Access Control (MAC) address is a unique identifier assigned to a Network Interface Controller (NIC) for use as a network address in communications within a network segment. Unlike IP addresses, which are logical and can change, MAC addresses are physical, hard-coded into the hardware by the manufacturer, and are essential for the Data Link Layer of the OSI model.

Solution: 1. **Understanding the Address Structure:** A MAC address is traditionally represented as six groups of two hexadecimal digits (e.g., `00:1A:2B:3C:4D:5E`).

2. **Calculating the Bit Count:** - Each hexadecimal digit represents 4 bits. - Each group (two digits) represents 8 bits (1 byte). - Since there are 6 groups, the total number of bits is $6 \times 8 = 48$ bits.

3. **Comparing with IP Addresses:** - IPv4 addresses are 32 bits long. - IPv6 addresses are 128 bits long. - MAC addresses sit in between at 48 bits.

4. **Role in Networking:** The first 24 bits identify the manufacturer (OUI), and the last 24 bits are the serial number assigned by that manufacturer. This 48-bit globally unique ID ensures that no two devices in the world (theoretically) share the same physical address.

5. **Final Deduction:** Based on the standard Ethernet and IEEE 802.3 specifications, the MAC address is precisely 48 bits.

Final Answer: 48

Answer: (C)

Q15.

Solution

Concept: The OSI (Open Systems Interconnection) model divides network communication into seven layers. Different networking devices operate at different layers based on their functionality. The Physical Layer (Layer 1) deals with the transmission and reception of raw bitstreams over a physical medium. It does not understand addressing or data packets; it only manages electrical, optical, or radio signals.

Solution: 1. **Evaluating Router:** Routers operate at the Network Layer (Layer 3). They use IP addresses to route packets between different networks.

2. **Evaluating Switch/Bridge:** Switches and Bridges operate at the Data Link Layer (Layer 2). They use MAC addresses to forward frames within a single network.

3. **Evaluating Repeater:** A repeater is a simple electronic device that receives a signal and retransmits it. It is used to overcome signal attenuation (loss of strength) over long distances. It operates entirely at the Physical Layer (Layer 1) because it does not inspect the data; it only works with the signal/bits.

4. **Step-by-Step Logic:** - Does the device look at IP addresses? No, so it's not Layer 3. - Does it look at MAC addresses? No, so it's not Layer 2. - Does it simply "regenerate" the signal to extend the distance? Yes, this is the definition of a Repeater.

5. **Conclusion:** The Repeater is the classic example of a Physical layer device designed for signal regeneration.

Final Answer: Repeater

Answer: (D)

Q16.

Solution

Concept: The Stack is a linear data structure that follows the LIFO (Last-In, First-Out) principle. This means that the last element added to the stack is the first one to be removed. Imagine a stack of dinner plates; you place plates one on top of the other, and when you need one, you take the one from the very top. In computer science, the operation to add an element is called 'Push', and the operation to remove an element is called 'Pop'.

Solution: 1. **Initial State:** The stack is empty.

2. **Step 1 (Push 'A'):** Element 'A' is added to the stack. The stack now contains: `[A]` (where 'A' is at the bottom).

3. **Step 2 (Push 'B'):** Element 'B' is added on top of 'A'. The stack now contains: `[A, B]`.

4. **Step 3 (Push 'C'):** Element 'C' is added on top of 'B'. The stack now contains: `[A, B, C]` (where 'C' is at the top).

5. **Step 4 (Popping):** When we start popping elements: - The first `pop()` operation removes the element at the top, which is 'C'. - The second `pop()` operation removes the next element at the top, which is 'B'. - The third `pop()` operation removes the last remaining element, which is 'A'.

6. **Result:** The sequence of elements removed is C, then B, then A. This is the reverse of the input order, confirming the LIFO behavior.

Final Answer: C, B, A

Answer: (B)

Q17.

Solution

Concept: Converting an infix expression (where operators are between operands) to a postfix expression (where operators follow operands) is a classic application of Stacks. This process follows the rules of operator precedence (PEMDAS/BODMAS) and ensures that the order of operations is preserved without the need for parentheses.

Solution: 1. **Expression Breakdown:** The expression is $(A + B) \times (C - D)$.

2. **Handling the First Parenthesis** $(A + B)$**:** - We encounter '(', push to operator stack. - We encounter 'A', append to output: `A`. - We encounter '+', push to operator stack. - We encounter 'B', append to output: `AB`. - We encounter ')', pop operators until '(' is found. Pop '+'. - Postfix so far: `AB+`.

3. **Handling the Operator** $\times$**:** - We encounter '$\times$', push to operator stack.

4. **Handling the Second Parenthesis** $(C - D)$**:** - We encounter '(', push to operator stack. - We encounter 'C', append to output: `AB+C`. - We encounter '-', push to operator stack. - We encounter 'D', append to output: `AB+CD`. - We encounter ')', pop operators until '(' is found. Pop '-'. - Postfix so far: `AB+CD-`.

5. **Final Step:** Pop the remaining operator '$\times$' from the stack and append to output. - Final Result: `AB+CD-`$\times$.

6. **Conclusion:** This sequence ensures that A and B are added, C and D are subtracted, and then the two results are multiplied.

Final Answer: $AB + CD - \times$

Answer: (A)

Q18.

Solution

Concept: A Queue is a fundamental linear data structure used in scenarios like printer spooling, CPU scheduling, and handling website traffic. Unlike a Stack, it is an open-ended structure. It operates on the FIFO (First-In, First-Out) principle, meaning that the first element added to the queue will be the first one to be removed.

Solution: 1. **The FIFO Principle:** Imagine a queue at a ticket counter. The person who arrives first is served first and leaves first. In data structures, we insert elements at the "Rear" (Enqueue) and remove them from the "Front" (Dequeue).

2. **Evaluating Options:** - **LIFO (Last-In, First-Out):** This is the principle of a Stack. The most recent item is the first out. - **LILO (Last-In, Last-Out):** This is logically equivalent to FIFO. If the first one is the first out, then the last one must be the last out. - **FIFO (First-In, First-Out):** This is the standard, primary definition of a Queue. - **Random Access:** This refers to structures like Arrays or Lists where any element can be reached directly via index, which is not how a pure Queue operates.

3. **Practical Application:** If you add data items 10, 20, and 30 to a queue, they will be removed in the exact same order: 10, then 20, then 30.

4. **Conclusion:** The defining characteristic and principle of a standard queue data structure is First-In, First-Out.

Final Answer: FIFO

Answer: (C)

Q19.

Solution

Concept: Evaluating a postfix expression (Reverse Polish Notation) involves using a stack. Postfix notation is advantageous for computers because it eliminates the need for parentheses and clearly defines the order of operations. When we encounter an operand, we push it onto the stack. When we encounter an operator, we pop the top two operands, apply the operator, and push the result back.

Solution: 1. **Scanning the Expression:** $10, 2, \times, 5, +, 3, -$

2. **Step-by-Step Processing:** - Read `10`: Push to stack. Stack: `[10]` - Read `2`: Push to stack. Stack: `[10, 2]` - Read $\times$: Pop `2` and `10`. Calculate $10 \times 2 = 20$. Push `20`. Stack: `[20]` - Read `5`: Push to stack. Stack: `[20, 5]` - Read `+`: Pop `5` and `20`. Calculate $20 + 5 = 25$. Push `25`. Stack: `[25]` - Read `3`: Push to stack. Stack: `[25, 3]` - Read `-`: Pop `3` and `25`. Calculate $25 - 3 = 22$. Push `22`. Stack: `[22]`

3. **Final Result:** After scanning the entire expression, the value remaining on the stack is the final answer. In this case, it is 22.

4. **Verification:** The equivalent infix expression would be $((10 \times 2) + 5) - 3$, which also equals 22.

Final Answer: 22

Answer: (A)

Q20.

Solution

Concept: In Python, lists are dynamic arrays that provide several methods for element removal. The `pop()` method is unique because it removes an element at a given index and also "returns" that value. If no index is specified, it removes the last element. Understanding list indexing (starting at 0) is crucial here.

Solution: 1. **Initial List:** `L = [10, 20, 30, 40, 50]`

2. **Indexing the List:** - Index 0: 10 - Index 1: 20 - Index 2: 30 - Index 3: 40 - Index 4: 50

3. **Executing pop(2):** - The command `L.pop(2)` targets the element at index 2. - The element at index 2 is `30`. - The method removes `30` from the list `L`. - Crucially, the `pop()` method returns the value it just removed.

4. **Result Analysis:** - The question asks for the "result of `L.pop(2)`", which refers to the return value of the function call. - The return value is `30`. - If the question had asked "What is the value of L after the operation?", the answer would have been `[10, 20, 40, 50]`.

5. **Conclusion:** The specific value ejected and returned by the function is 30.

Final Answer: 30

Answer: (B)

Q21.

Solution

Concept: To implement a Stack data structure in Python, we need to choose methods that mimic the "push" and "pop" behavior while maintaining the LIFO (Last-In, First-Out) principle efficiently. Python's built-in `list` is ideal for this because it allows for $O(1)$ amortized time complexity when adding or removing elements from the end of the structure.

Solution: 1. **The Push Operation:** In a stack, "pushing" means adding an element to the "top." In a Python list, the "top" is naturally represented by the end of the list (the highest index). The `append()` method adds an element to the end of the list, making it the perfect implementation for "Push."

2. **The Pop Operation:** In a stack, "popping" means removing the most recently added element from the "top." The Python list method `pop()` (without arguments) removes and returns the last element of the list. This perfectly matches the LIFO requirement.

3. **Evaluating Distractors:** - `insert()` and `delete()` are inefficient ($O(n)$) because they shift all other elements. - `push()` is not a built-in method for Python lists; it is a conceptual term. - `add()` is used for sets, and `remove()` searches for a value rather than using an index or position.

4. **Conclusion:** The idiomatic and most efficient way to use a list as a stack in Python is by using `append()` to add elements and `pop()` to remove them.

Final Answer: append() and pop()

Answer: (C)

Q22.

Solution

Concept: Time complexity analysis (Big O notation) is used to describe the efficiency of an algorithm or operation as the input size (n) grows. Python lists are implemented as dynamic arrays, which means they store elements in contiguous memory locations. This physical structure allows for direct calculation of the memory address of any element if its index is known.

Solution: 1. **Mechanism of Indexing:** When you access `L[i]`, the computer does not scan the list from the beginning. Instead, it uses a simple mathematical formula: `Address = Base_Address + (index * size_of_element)`.

2. **Execution Speed:** Because this is a single calculation followed by a memory fetch, the time it takes to access the 1^{st} element is exactly the same as the time it takes to access the $1,000,000^{th}$ element.

3. **Complexity Definition:** An operation that takes a constant amount of time regardless of the size of the data structure is said to have a time complexity of $O(1)$, or constant time.

4. **Comparison:** - $O(n)$ would mean we have to search the whole list (like finding a value). - $O(\log n)$ is typical for binary search on sorted data. - $O(n^2)$ is typical for nested loops.

5. **Conclusion:** Accessing an element by index in a dynamic array (Python list) is a high-speed, $O(1)$ operation.

Final Answer: $O(1)$

Answer: (A)

Q23.

Solution

Concept: Reversing data is one of the most common applications of a Stack. Because a Stack follows the Last-In, First-Out (LIFO) principle, any sequence of items pushed into a stack will come out in the exact reverse order when popped. This makes the Stack uniquely suited for string reversal compared to other linear data structures like Queues or Linked Lists.

Solution: 1. **Step-by-Step Logic:** To reverse the string "PYTHON": - Push 'P', then 'Y', then 'T', then 'H', then 'O', then 'N' into the stack. - The stack top is now 'N'. - Pop the elements one by one. The first to come out is 'N', then 'O', then 'H', then 'T', then 'Y', and finally 'P'. - The resulting sequence is "NOHTYP", which is the reverse of the original.

2. **Comparison with Queue:** A Queue (FIFO) would return the elements in the same order they were entered ("PYTHON"), making it useless for reversal.

3. **Comparison with Linked List:** While you can reverse a linked list, it requires pointer manipulation or recursion, which is more complex than the simple push/pop logic of a stack.

4. **Conclusion:** Due to its inherent LIFO nature, the Stack is the most natural and efficient data structure for the purpose of reversing sequences.

Final Answer: Stack

Answer: (B)

Q24.

Solution

Concept: A circular queue is an advanced version of a regular queue where the last position is connected back to the first position to make use of empty spaces created by deletions. The state of the queue (empty, full, or containing elements) is determined by the relationship between the `Front` and `Rear` pointers.

Solution: 1. **Pointer Definitions:** - `Front`: Points to the element to be removed. - `Rear`: Points to the last element added.

2. **Queue Condition Analysis:** - Total Size (N) = 5. - `Front` = 3. - `Rear` = 2.

3. **Testing for "Full" State:** In a circular queue, the "Full" condition is often checked using: `(Rear + 1) % Size == Front`. - Let's calculate: `(2 + 1) % 5 = 3 % 5 = 3`. - Since the result (3) is equal to the `Front` (3), the queue has no more space for new elements.

4. **Visualization:** - Index 3: Occupied - Index 4: Occupied - Index 0: Occupied - Index 1: Occupied - Index 2: Occupied (Rear) - The next available spot would be index 3, but that is where the Front is. Thus, the queue is full.

Final Answer: Full

Answer: (B)

Q25.

Solution

Concept: In stack implementation, we use a variable (often called `top`) to keep track of the index of the uppermost element. Since standard programming indices start at 0, the value of this pointer provides immediate information about the number of elements in the stack and whether operations like 'Pop' are even possible.

Solution: 1. **Pointer States:** - If `top = 0`, the stack has one element (at index 0). - If `top = n-1`, the stack is full. - If `top = -1`, it means there are no valid indices currently holding data.

2. **Underflow vs. Overflow:** - **Underflow** occurs when you try to pop an element from an empty stack. If `top` is -1, any pop operation will result in an error because there is nothing to remove. - **Overflow** occurs when you try to push into a stack that is already at its maximum capacity.

3. **Logic Check:** A pointer value of -1 is the standard convention to represent an empty stack. It serves as a guard against "Underflow" errors.

4. **Conclusion:** When the 'Top' pointer is at -1, the stack is empty, and this state is synonymous with the condition that leads to Underflow if a removal is attempted.

Final Answer: Underflow

Answer: (B)

Q26.

Solution

Concept: A Queue is a linear data structure designed to manage data in a sequential manner, adhering to the First-In, First-Out (FIFO) principle. This structure is analogous to a line of people waiting for a service. To maintain this specific order, a Queue requires two primary operations: one for adding elements and one for removing them. Standard terminology is used to distinguish these operations from those used in Stacks (Push/Pop).

Solution: 1. **Defining the Insertion Operation:** The process of adding a new element to the back of a queue is formally known as `Enqueue`. This operation increases the size of the queue and moves the 'Rear' pointer to the new element.

2. **Defining the Removal Operation:** The process of removing an element from the front of the queue is known as `Dequeue`. This operation follows the FIFO rule, ensuring the oldest element is processed first.

3. **Analyzing the Options:** - **Push:** This is the term for inserting into a Stack. - **Pop:** This is the term for removing from a Stack. - **Dequeue:** This is the term for removing from a Queue. - **Enqueue:** This is the term for inserting into a Queue.

4. **Logic Check:** In many programming libraries, these names are strictly followed. For instance, in many implementations, calling `enqueue(item)` adds the item to the rear, while `dequeue()` removes it from the front.

5. **Conclusion:** For the specific task of inserting a new element into a queue, `Enqueue` is the correct technical term.

Final Answer: Enqueue

Answer: (C)

Q27.

Solution

Concept: Binary Search is a highly efficient searching algorithm that works on the ”divide and conquer” principle. It repeatedly halves the search interval. For this algorithm to work, the data must be sorted. The time complexity of Binary Search is $O(\log_2 n)$, which means the number of comparisons grows very slowly as the number of elements increases.

Solution: 1. **The Mathematical Formula:** The maximum number of comparisons required for Binary Search is given by the formula: $\lceil \log_2(n) \rceil$, where n is the number of elements.

2. **Calculation for n = 128:** - We need to find the power to which 2 must be raised to equal 128. - $2^1 = 2$ - $2^2 = 4$ - $2^3 = 8$ - $2^4 = 16$ - $2^5 = 32$ - $2^6 = 64$ - $2^7 = 128$

3. **Deduction:** Since $2^7 = 128$, the value of $\log_2(128)$ is exactly 7.

4. **Interpreting the Result:** In the worst-case scenario (where the element is at the very last possible split or not in the list at all), you will have to perform 7 divisions/comparisons to narrow the search space down to a single element.

5. **Comparison:** A Linear Search would require up to 128 comparisons for the same task. This demonstrates why Binary Search is preferred for large, sorted datasets.

Final Answer: 7

Answer: (C)

Q28.

Solution

Concept: Sorting algorithms are categorized based on how they move and compare elements. Selection Sort is an ”in-place” comparison sort. It is known for its simplicity and the fact that it performs a minimal number of swaps compared to algorithms like Bubble Sort. It logically divides the array into two parts: a sorted subarray and an unsorted subarray.

Solution: 1. **Mechanism of Selection Sort:** - The algorithm starts at the first position of the array. - it scans the entire remaining unsorted portion to find the absolute minimum value. - Once found, it swaps this minimum value with the element at the current starting position. - The boundary of the sorted part moves one step to the right.

2. **Step-by-Step Example:** - Array: $[5, 3, 8, 2]$ - Pass 1: Find min in $[5, 3, 8, 2]$, which is 2. Swap with index 0. Array: $[2, 3, 8, 5]$. - Pass 2: Find min in $[3, 8, 5]$, which is 3. It’s already in place. - Pass 3: Find min in $[8, 5]$, which is 5. Swap with index 2. Array: $[2, 3, 5, 8]$.

3. **Evaluating Options:** - **Bubble Sort:** Compares adjacent elements and swaps them. - **Insertion Sort:** Picks an element and inserts it into its correct position in a sorted prefix. - **Selection Sort:** Repeatedly ”selects” the minimum element and moves it to the front.

4. **Conclusion:** The description ”repeatedly finds the minimum element and puts it at the beginning” is the textbook definition of Selection Sort.

Final Answer: Selection Sort

Answer: (C)

Q29.

Solution

Concept: Bubble Sort is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. This process is repeated until the list is sorted. It is named "Bubble Sort" because larger elements "bubble up" to their correct positions at the end of the list with each pass.

Solution: 1. **The First Pass Mechanism:** During the first pass, the algorithm compares elements at index 0 and 1, then 1 and 2, and so on, until it reaches the last pair $(n-2, n-1)$.

2. **Tracing the Largest Element:** - No matter where the largest element is located initially, the adjacent comparisons will eventually move it to the right. - For example, in $[3, 9, 2, 5]$, 9 is compared with 2 and swapped ($[3, 2, 9, 5]$), then 9 is compared with 5 and swapped ($[3, 2, 5, 9]$).

3. **Result of the Pass:** By the end of the very first pass, the absolute largest element in the array is guaranteed to have "bubbled up" to the final index $(n-1)$.

4. **Subsequent Passes:** The second pass will place the second-largest element at index $n-2$, and so on. This is why the inner loop of Bubble Sort often decreases its range by 1 after each outer loop iteration.

5. **Conclusion:** After the first pass, the element at the **last position** (the maximum value) is correctly placed.

Final Answer: The last element

Answer: (B)

Q30.

Solution

Concept: Linear Search (also known as Sequential Search) is the most basic searching algorithm. It examines each element of a list one by one until a match is found or the end of the list is reached. While its average and worst-case complexities are $O(n)$, the best-case complexity describes the scenario where the algorithm performs the minimum possible work.

Solution: 1. **Defining the Best Case:** The best-case scenario for any search algorithm occurs when the target element is found in the very first location checked.

2. **Executing the Search:** - Input: Array L of size n, Target X. - The algorithm checks $L[0]$. - If $L[0] == X$, the search terminates immediately.

3. **Calculating Complexity:** - In this specific instance, the algorithm performed exactly 1 comparison. - The number of operations did not depend on the size of the array (n). Whether the array had 10 elements or 10 million elements, if the target was at index 0, it only took 1 step.

4. **Big O Notation:** Constant time, where the performance is independent of the input size, is denoted as $O(1)$.

5. **Comparison:** The worst case $O(n)$ occurs if the element is at the end or not present at all, requiring n comparisons.

Final Answer: $O(1)$

Answer: (A)

Q31.

Solution

Concept: Insertion Sort is an intuitive sorting algorithm that builds the final sorted array one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort. However, it provides several advantages, such as being easy to implement and efficient for small data sets or data sets that are already substantially sorted. Its logic is frequently compared to the manual process of sorting a hand of playing cards.

Solution: 1. **The Card Sorting Analogy:** Imagine you are dealt cards one by one. You hold the cards you have already received in a sorted order in your left hand. When you receive a new card, you ”insert” it into the correct position in your left hand by comparing it with the cards already there, shifting the larger cards to the right to make space.

2. **Algorithmic Steps:** - We start from the second element (index 1) and assume the first element is a ”sorted” subarray of size 1. - We pick the current element and compare it with the elements in the sorted subarray to its left. - If the element in the sorted part is greater than our current element, we shift it to the right. - We repeat this until we find the correct spot and insert the element there.

3. **Comparison with other algorithms:** - **Selection Sort** finds the minimum and swaps. - **Bubble Sort** swaps adjacent pairs. - **Insertion Sort** ”inserts” into a sorted portion, exactly like sorting cards.

4. **Conclusion:** Because of this specific ”pick and shift” mechanism that mirrors manual card sorting, Insertion Sort is the definitive answer.

Final Answer: Insertion Sort

Answer: (C)

Q32.

Solution

Concept: Binary Search is a "Divide and Conquer" algorithm used to find the position of a target value within a collection. Its efficiency comes from the fact that it eliminates half of the search space in every step. However, this efficiency relies on a strict mathematical requirement regarding the organization of the data. If this requirement is not met, the logic of comparing the "middle" element to the target becomes fundamentally flawed.

Solution: 1. **The Logic of Binary Search:** The algorithm looks at the middle element (M) of the current range. If the target (T) is greater than M, the algorithm assumes T must be in the upper half. If T is less than M, it assumes T is in the lower half.

2. **The Necessity of Order:** This assumption—that "greater than" implies a specific direction—only holds true if the data is **Sorted**.

3. **Example of Failure:** Consider the unsorted list $[10, 50, 20, 40, 30]$. If we search for 20, the middle is 20. But if we search for 30, and the list was $[10, 30, 50, 20]$, the middle is 30 or 50. Without a sorted structure, we cannot know which direction to turn, forcing us back to a Linear Search ($O(n)$).

4. **Types of Sort:** The data can be in ascending or descending order, but it must be consistently sorted.

5. **Conclusion:** Binary Search requires the data structure to be in sorted order to function with its $O(\log n)$ efficiency.

Final Answer: Sorted

Answer: (B)

Q33.

Solution

Concept: In computer science, a sorting algorithm is considered ”stable” if it preserves the relative order of records with equal keys (values). For example, if two records with the same value appear in the input at indices i and j where $i < j$, a stable sort ensures they appear in the output in that same relative order. An ”unstable” sort does not guarantee this preservation.

Solution: 1. **Evaluating Bubble Sort:** Bubble sort only swaps adjacent elements if they are strictly in the wrong order ($A > B$). If two elements are equal, no swap occurs. Thus, their relative order is maintained. It is a **Stable** sort.

2. **Evaluating Insertion Sort:** In insertion sort, an element is moved past others only if it is strictly smaller (or larger, depending on order). Equal elements are not moved past each other. It is a **Stable** sort.

3. **Evaluating Selection Sort:** Selection sort works by finding the minimum and swapping it with the element at the current starting position. This ”long-distance” swap can move an element past an identical one elsewhere in the array, potentially changing their relative order.

- **Example:** In $[2a, 2b, 1]$, Selection Sort finds 1 as the minimum and swaps it with $2a$. The result is $[1, 2b, 2a]$. Note that $2a$ now comes after $2b$. The stability is lost.

4. **Conclusion:** Selection Sort is fundamentally unstable because of its swapping logic.

Final Answer: Selection Sort

Answer: (C)

Q34.

Solution

Concept: Time complexity defines how the execution time of an algorithm grows relative to the input size n. Selection Sort is known for its consistent performance; regardless of whether the initial array is sorted, reverse-sorted, or completely random, the algorithm always performs the same number of comparisons to find the minimum element in each pass.

Solution: 1. **Algorithmic Analysis:** - In the first pass, we make $(n-1)$ comparisons. - In the second pass, we make $(n-2)$ comparisons. - This continues until the last pass, which involves 1 comparison.

2. **Summation:** The total number of comparisons is the sum of the first $(n-1)$ integers:

$$\text{Total Comparisons} = \frac{(n-1) \times n}{2} = \frac{n^2 - n}{2}$$

3. **Asymptotic Growth:** In Big O notation, we drop the lower-order terms and constants. The n^2 term dominates the growth of the function.

4. **Best, Average, and Worst Case:** Unlike Bubble Sort or Insertion Sort, which can have an $O(n)$ best case if the list is already sorted, Selection Sort *always* scans the entire unsorted portion to confirm it has the minimum. Thus, its complexity is $O(n^2)$ in all cases, including the average case.

5. **Conclusion:** The average case time complexity of Selection Sort is squarely $O(n^2)$.

Final Answer: $O(n^2)$

Answer: (B)

Q35.

Solution

Concept: Bubble Sort operates by performing multiple passes over the array. Each pass aims to move (or ”bubble”) the largest remaining unsorted element to its correct position. The number of passes required to guarantee that an array is fully sorted is a key property of the algorithm’s worst-case performance.

Solution: 1. **General Rule for Passes:** For an array of n elements, Bubble Sort requires at most $(n - 1)$ passes to sort the entire array. This is because each pass places at least one element in its correct final position.

2. **Logical Deduction:** After $(n - 1)$ elements are correctly placed in their final positions (the largest, second largest, etc.), the n^{th} element (the smallest) has no choice but to be in its correct position (the first index). Therefore, n passes are never necessary; $(n - 1)$ is sufficient.

3. **Applying to the Problem:** - The number of elements n is 6. - Following the rule, the number of passes required is $6 - 1 = 5$.

4. **Detailed Steps:** - Pass 1: Largest element is placed. - Pass 2: Second largest is placed. - Pass 3: Third largest is placed. - Pass 4: Fourth largest is placed. - Pass 5: Fifth largest is placed. - Result: All 6 elements are now in the correct order.

5. **Conclusion:** For a 6-element array, 5 passes are required.

Final Answer: 5

Answer: (B)

Q36.

Solution

Concept: In Python, exception handling is a mechanism to manage runtime errors, preventing the program from crashing abruptly. This is achieved using the `try-except` block. The code that might cause an error is placed in the `try` block, while the code to handle that specific error (should it occur) is placed in the `except` block.

Solution: 1. **The try block:** This block contains the code that the program attempts to execute. For example, dividing by zero or opening a non-existent file.

2. **The except block:** If an error occurs within the `try` block, Python stops execution of that block and jumps immediately to the `except` block. This block ”catches” the exception.

3. **Evaluating Options:** - **catch:** This keyword is used in languages like Java, C++, and C#, but not in Python. - **except:** This is the official Python keyword used to handle exceptions. - **throw/error:** These are not keywords used for catching exceptions in Python; `raise` is used to throw an exception. - **try:** While used, the question specifically asks for the block that ”handles” the exception, which is the role of `except`.

4. **Logic Check:** A standard Python error-handling structure looks like this:

```
try:
    # code
except ValueError:
    # handle error
```

5. **Conclusion:** The `except` block is the specific component used to catch and manage exceptions in Python.

Final Answer: except

Answer: (B)

Q37.

Solution

Concept: Python's `try` statement can have an optional `finally` clause. The primary purpose of this block is to define "clean-up" actions that must be executed under all circumstances. This is particularly important for releasing external resources, such as closing file streams or terminating database connections, regardless of whether the operation was successful or not.

Solution: 1. **Scenario A (No Exception):** If the code in the `try` block runs successfully, the `except` block is skipped, but the `finally` block still executes before the program continues.

2. **Scenario B (Exception Occurs):** If an error occurs, the `except` block runs to handle it. After the `except` block completes, the `finally` block executes.

3. **Scenario C (Unhandle Exception):** Even if an exception occurs that isn't caught by an `except` block, the `finally` block will execute before the program terminates.

4. **Evaluating Options:** - Option (A) and (B) are incorrect because they imply conditional execution. - Option (C) states it executes "Always, regardless of an exception," which is the defining characteristic of `finally`.

5. **Final Deduction:** The execution of the `finally` block is guaranteed as long as the Python interpreter is running.

Final Answer: Always, regardless of an exception

Answer: (C)

Q38.

Solution

Concept: File handling in Python allows for different modes of opening files. There are two main categories: Text mode (default) and Binary mode. Binary mode is essential when dealing with non-text files like images, audio, or serialized Python objects (using pickle), where data must be read exactly as bits without any encoding/decoding.

Solution: 1. **Basic Mode Characters:** - `'r'`: Read (Text mode) - `'w'`: Write (Text mode) - `'a'`: Append (Text mode)

2. **Binary Mode Specifier:** To signal binary mode, the character `'b'` must be appended to the base mode character.

3. **Combining Modes:** - To read a text file, we use `'r'`. - To read a binary file, we combine `'r'` and `'b'` to get `'rb'`.

4. **Step-by-Step Logic:** - Does the user want to read? Yes (`'r'`). - Does the user want binary? Yes (`'b'`). - Result: `'rb'`.

5. **Conclusion:** The mode string `'rb'` is the standard Python notation for "read binary."

Final Answer: 'rb'

Answer: (B)

Q39.

Solution

Concept: When writing to text files in Python, we often have data stored in different formats. Python's file objects provide specific methods to handle single strings versus sequences of strings. Understanding which method to use prevents errors and ensures that lists of data are written efficiently to the storage medium.

Solution: 1. **Analyzing write():** The `write(string)` method expects a single string as its argument. If you pass a list of strings to it, Python will raise a `TypeError`.

2. **Analyzing writelines():** The `writelines(list)` method is designed specifically to take an iterable (like a list or tuple) and write each element to the file sequentially. It does not automatically add newlines; it simply writes the elements of the list back-to-back.

3. **Analyzing other options:** - `putlines()` is not a standard Python file method. - `dump()` is a method found in the `pickle` or `json` modules, used for serialization, not for writing a plain list of strings to a text file.

4. **Step-by-Step Logic:** - Input: `["Line 1", "Line 2", "Line 3"]` - Task: Write to text file. - Choice: `file_object.writelines(list_name)`.

5. **Conclusion:** For a list of strings, `writelines()` is the correct and most direct method.

Final Answer: writelines()

Answer: (B)

Q40.

Solution

Concept: The `with` statement in Python is used for resource management and is often referred to as a "context manager." It simplifies file handling by ensuring that "setup" and "teardown" operations are handled automatically. This is considered the "Pythonic" way to handle files because it makes the code cleaner and more robust against leaks.

Solution: 1. **Traditional File Handling:** Previously, one would use `f = open(...)` and then had to remember to call `f.close()` at the end. If an error occurred before the `close()` line, the file might remain open in memory, leading to corrupted data or resource exhaustion.

2. **The with Advantage:** When using `with open(...) as f:`, Python creates a context. As soon as the block of code inside the `with` statement is finished (or if an exception occurs), Python automatically calls the `.close()` method on the file object.

3. **Evaluating the Options:** - It does not necessarily run faster. - It does not encrypt files. - While it can create a file (if in 'w' mode), that is a feature of `open()`, not specifically the `with` statement. - The primary and most significant advantage is the automatic closing of the file.

4. **Final Deduction:** The `with` statement is used to guarantee that resources are cleaned up properly without manual intervention.

Final Answer: It automatically closes the file

Answer: (B)

Q41.

Solution

Concept: Serialization in Python is the process of converting a Python object (like a list, dictionary, or custom class) into a byte stream so that it can be saved to a file or transmitted over a network. This is also known as "pickling." Conversely, "unpickling" is the process of converting that byte stream back into a Python object. Python provides a dedicated built-in module for this purpose.

Solution: 1. **Module Analysis:** - `os`: Used for interacting with the operating system (e.g., handling paths, creating directories). - `math`: Provides mathematical functions like `sin`, `cos`, and `sqrt`. - `sys`: Provides access to variables and functions that interact with the Python interpreter. - `pickle`: This is the specific module designed for serializing and de-serializing Python objects.

2. **Functionality:** The `pickle` module provides two primary functions: - `pickle.dump(obj, file)`: Writes the serialized object to a binary file. - `pickle.load(file)`: Reads the serialized object from a binary file and restores it.

3. **Binary Mode Requirement:** It is important to note that when using the `pickle` module, the file must always be opened in binary mode (`'wb'` for writing or `'rb'` for reading).

4. **Conclusion:** The `pickle` module is the standard tool for binary file input/output operations involving Python-specific data structures.

Final Answer: pickle

Answer: (C)

Q42.

Solution

Concept: File pointers (or "handles") keep track of the current position within an open file where the next read or write operation will occur. Python provides the `seek()` method to manually move this pointer to a specific byte location. The method takes two arguments: `offset` (number of bytes) and `whence` (reference point).

Solution: 1. **Understanding the Reference Point (whence):** - `0`: Relative to the beginning of the file (Default). - `1`: Relative to the current pointer position. - `2`: Relative to the end of the file.

2. **Applying the Command:** The method call is `seek(5, 0)`. - The second argument is `0`, which means the starting point is the very beginning of the file. - The first argument is `5`, which means move forward 5 bytes.

3. **Detailed Result:** After this command is executed, the file pointer will sit at the 6^{th} byte (offset 5 from 0). Any subsequent `read()` or `write()` call will start from this position.

4. **Logic Check:** - Option (A) describes `seek(5, 1)`. - Option (B) describes `seek(-5, 2)` (usually used with negative offsets). - Option (C) accurately describes `seek(5, 0)`.

5. **Conclusion:** `seek(5, 0)` moves the pointer to a fixed position 5 bytes from the start of the file.

Final Answer: Moves the pointer 5 bytes from the beginning of the file

Answer: (C)

Q43.

Solution

Concept: Python has a hierarchy of built-in exceptions that help developers identify exactly what went wrong during code execution. For File I/O operations, several things can go wrong: permission issues, full disks, or missing files. Python updated its exception hierarchy in version 3.3 to make these errors more specific and easier to catch.

Solution: 1. **Identifying the Scenario:** The code attempts to open a file for reading using `open('filename', 'r')`, but the operating system cannot find a file with that name in the specified directory.

2. **Analyzing the Exception:** - `IOError`: This was the general base class for input/output errors in older versions of Python. While still present, it is now an alias for `OSError`. - `FileNotFoundError`: This is a specific subclass of `OSError` that is raised exactly when a file or directory is requested but does not exist. - `ValueError`: Raised when a function receives an argument of the right type but inappropriate value (e.g., `int('abc')`). - `ImportError`: Raised when a module cannot be found.

3. **Step-by-Step Logic:** When `open()` fails because the path is invalid, Python checks the reason for failure and raises the most specific exception available, which is `FileNotFoundError`.

4. **Conclusion:** For a missing file during a read operation, `FileNotFoundError` is the specific exception raised.

Final Answer: FileNotFoundError

Answer: (B)

Q44.

Solution

Concept: Protocols are sets of rules that govern how data is transmitted over a network. Different protocols are optimized for different tasks, such as sending emails, browsing websites, or transferring raw files. File Transfer Protocol (FTP) is a standard communication protocol used to transfer computer files from one host to another over a TCP-based network, such as the Internet.

Solution: 1. **Comparing Protocols:** - **HTTP (Hypertext Transfer Protocol)**: Used primarily for fetching HTML documents and web assets (images, scripts). While it can transfer files, its primary purpose is web browsing. - **SMTP (Simple Mail Transfer Protocol)**: Used for sending emails from a client to a server. - **FTP (File Transfer Protocol)**: Specifically designed for bulk file movement. It handles the complexities of directory listing, file permissions, and binary/text transfer modes. - **POP3 (Post Office Protocol v3)**: Used by email clients to retrieve emails from a mail server.

2. **Key Characteristics of FTP:** It typically uses two separate connections between the client and the server: a command channel (port 21) for control information and a data channel (port 20) for the actual file content.

3. **Conclusion:** Because its core architectural purpose is the movement of files between a client and a server, FTP is the correct answer.

Final Answer: FTP

Answer: (C)

Q45.

Solution

Concept: Switching techniques define how data is routed through a network of intermediate nodes. The three main types are Circuit Switching, Message Switching, and Packet Switching. Circuit switching is the oldest method and is most commonly associated with traditional telephone networks (PSTN).

Solution: 1. **Mechanism of Circuit Switching:** Before any data or voice is sent, a physical connection (circuit) is established from the sender to the receiver. This path is dedicated and reserved for the entire duration of the communication.

2. **Comparison with other methods:** - **Packet Switching**: Data is broken into small packets that may take different paths and are reassembled at the destination (used by the Internet). No dedicated path is reserved. - **Message Switching**: The entire message is sent from one node to the next (store-and-forward) until it reaches the destination.

3. **Advantages and Disadvantages:** - **Advantage**: Once the path is set, there is a fixed bit rate and no delay variation (jitter). - **Disadvantage**: It is inefficient because the resources are locked even if no data is being sent (idle time).

4. **Logic Check:** The phrase ”dedicated physical path established before data transfer” is the defining requirement of Circuit Switching.

Final Answer: Circuit Switching

Answer: (C)

Q46.

Solution

Concept: Malware is a broad term for malicious software designed to disrupt, damage, or gain unauthorized access to a computer system. Viruses and Worms are two of the most common types, and they are often confused. The key distinction lies in how they replicate and whether they require a ”host” to function.

Solution: 1. **Defining a Virus:** A computer virus is a type of malware that, when executed, replicates itself by modifying other computer programs and inserting its own code. Just like a biological virus, it needs a ”host” (a legitimate program or file) to survive and spread. When the infected program is run, the virus code runs too.

2. **Comparing with others:** - **Worm**: A standalone program that replicates itself to spread to other computers, often using a network. It does **not** need to attach to a host program. - **Trojan Horse**: A program that appears useful but contains hidden malicious functions. It does not replicate itself. - **Spyware**: Software that secretly monitors user activity.

3. **Detailed Reasoning:** The question specifies that the program ”attaches itself to another program” and ”replicates when the host program is executed.” This perfectly matches the behavior of a Virus.

Final Answer: Virus

Answer: (B)

Q47.

Solution

Concept: Spoofing is a type of cyberattack in which someone or something pretends to be something else in an attempt to gain our confidence, get access to our systems, or steal data. It can occur at many levels, including IP spoofing, Email spoofing, and DNS spoofing. The core goal is deception through identity theft.

Solution: 1. **Defining Spoofing:** In the context of network security, spoofing occurs when a person or program successfully identifies as another by falsifying data. For example, a hacker might send an IP packet with a forged source IP address to make it look like the packet came from a trusted internal machine.

2. **Evaluating Options:** - **Phishing**: A technique used to gather sensitive info (like passwords) using deceptive emails. While it uses spoofing, the ”pretending to be someone else” part of the identity is specifically Spoofing. - **Eavesdropping**: Intercepting data as it travels over the network (listening). - **Denial of Service (DoS)**: Flooding a system with traffic to make it unavailable.

3. **Step-by-Step Logic:** - The attacker wants access. - The attacker disguises their identity as a trusted user. - This identity falsification is Spoofing.

4. **Conclusion:** Pretending to be a trusted entity to gain unauthorized access is the definition of spoofing.

Final Answer: Spoofing

Answer: (C)

Q48.

Solution

Concept: A Firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules. It establishes a barrier between a trusted internal network and untrusted external networks, such as the Internet. Firewalls can be hardware-based, software-based, or a combination of both.

Solution: 1. **Primary Function:** The main job of a firewall is to act as a ”gatekeeper.” It inspects every packet of data that tries to enter or leave the network. If the packet doesn’t meet the ”safety criteria” (rules), the firewall blocks it.

2. **Evaluating Distractors:** - It does not speed up the connection; in fact, heavy inspection can slightly slow it down. - It is not a storage device for backups. - It is not a wireless access point (though some home routers include both).

3. **Detailed Mechanism:** Firewalls use ”Packet Filtering,” ”Stateful Inspection,” or ”Proxy Services” to identify threats like unauthorized access attempts, malware, or suspicious traffic patterns.

4. **Conclusion:** The monitoring and control of network traffic is the fundamental reason for a firewall’s existence.

Final Answer: To monitor and control incoming/outgoing network traffic

Answer: (B)

Q49.

Solution

Concept: Security on the World Wide Web is achieved through encryption protocols. Standard HTTP transmits data in plain text, which can be intercepted by anyone on the same network. HTTPS (Hypertext Transfer Protocol Secure) adds a layer of security by using TLS (Transport Layer Security) or its predecessor, SSL (Secure Sockets Layer), to encrypt the communication between the browser and the server.

Solution: 1. **Understanding HTTPS:** The ’S’ at the end stands for ”Secure.” When you visit an HTTPS site, your browser and the server perform a ”handshake” to establish an encrypted connection.

2. **Identifying Visual Cues:** Modern browsers display a small padlock icon in the address bar to indicate that an HTTPS connection is active.

3. **Comparing Protocols:** - **HTTP**: Unencrypted, vulnerable to eavesdropping. - **HTTPS**: Encrypted, ensures data integrity and privacy. - **IP/TCP**: Foundation protocols for moving data, but they do not provide user-level encryption by themselves.

4. **Conclusion:** HTTPS is the standard protocol for securing web data and providing visual confirmation to users via the padlock icon.

Final Answer: HTTPS

Answer: (B)

Q50.

Solution

Concept: Phishing is a specific type of social engineering attack often used to steal user data, including login credentials and credit card numbers. It is characterized by its reliance on psychological manipulation rather than technical vulnerabilities. Phishing attacks typically use email or text messages to trick the victim into performing an action.

Solution: 1. **How Phishing Works:** An attacker sends an email that looks like it's from a bank, Netflix, or a government agency. The email usually contains a "link" that takes the user to a fake website that looks identical to the real one. Once the user enters their details on the fake site, the attacker captures them.

2. **Evaluating Options:** - **Hacking**: A broad term for gaining unauthorized access. - **Spamming**: Sending unsolicited bulk messages, usually for advertising. It is annoying but not necessarily a theft attempt. - **Cracking**: Specifically refers to breaking software protections or passwords. - **Phishing**: The specific act of using fraudulent emails to steal personal information.

3. **Final Deduction:** The description of "fraudulent emails appearing to be from reputable companies" is the exact definition of a phishing campaign.

Final Answer: Phishing

Answer: (B)

Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	D	2	B	3	A	4	B	5	B
6	B	7	D	8	A	9	B	10	C
11	B	12	B	13	C	14	C	15	D
16	B	17	A	18	C	19	A	20	B
21	C	22	A	23	B	24	B	25	B
26	C	27	C	28	C	29	B	30	A
31	C	32	B	33	C	34	B	35	B
36	B	37	C	38	B	39	B	40	B
41	C	42	C	43	B	44	C	45	C
46	B	47	C	48	B	49	B	50	B