Technical Interview Questions and Answers

Introduction and Basic Programming

1. **Write code to print the duplicate alphabet in a string**

```
public static void printDuplicateCharacters(String str) {
    Map<Character, Integer> charCountMap = new HashMap<>();

    // Count each character
    for (char c : str.toCharArray()) {
        charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);
    }

    // Print characters that appear more than once
    System.out.println("Duplicate characters in string '" + str + "':");
    for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
        if (entry.getValue() > 1) {
            System.out.println(entry.getKey() + " - appears " + entry.getValue() + " times");
        }
    }
}
```

2. **Write code to reverse a string**

```
// Using StringBuilder
public static String reverseString(String str) {
    return new StringBuilder(str).reverse().toString();
}

// Manual implementation
public static String reverseStringManual(String str) {
    char[] charArray = str.toCharArray();
    int left = 0;
    int right = charArray.length - 1;

    while (left < right) {
        // Swap characters
        char temp = charArray[left];
        charArray[left] = charArray[right];
        charArray[right] = temp;

        // Move indices
        left++;
        right--;
    }

    return new String(charArray);
}
```

3. **Write merge sort code**

```
public static void mergeSort(int[] arr, int left, int right) {
    if (left < right) {
        // Find middle point
        int mid = left + (right - left) / 2;

        // Sort first and second halves
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);

        // Merge the sorted halves
        merge(arr, left, mid, right);
    }
}

public static void merge(int[] arr, int left, int mid, int right) {
    // Find sizes of two subarrays to be merged
    int n1 = mid - left + 1;
    int n2 = right - mid;

    // Create temporary arrays
    int[] L = new int[n1];
    int[] R = new int[n2];

    // Copy data to temporary arrays
    for (int i = 0; i < n1; i++)
        L[i] = arr[left + i];
    for (int j = 0; j < n2; j++)
        R[j] = arr[mid + 1 + j];

    // Merge the temporary arrays
    int i = 0, j = 0;
    int k = left;

    while (i < n1 && j < n2) {
        if (L[i] <= R[j]) {
            arr[k] = L[i];
            i++;
        } else {
            arr[k] = R[j];
            j++;
        }
        k++;
    }

    // Copy remaining elements of L[] if any
    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
```

```
    }

    // Copy remaining elements of R[] if any
    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}
```

4. **Why do you use int before main?**

- int is the return type of the main method
- It indicates that the main method returns an integer value to the operating system upon completion
- By convention, returning 0 indicates successful execution, while non-zero values indicate errors
- The operating system can use this return value to determine if the program executed successfully
- Example: `int main() { /* code */ return 0; }`

5. **Taking a user input of string in C++**

```
#include <iostream>
#include <string>

int main() {
    std::string userInput;

    // For single word
    std::cout << "Enter a word: ";
    std::cin >> userInput;
    std::cout << "You entered: " << userInput << std::endl;

    // Clear input buffer
    std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');

    // For a line with spaces
    std::cout << "Enter a sentence: ";
    std::getline(std::cin, userInput);
    std::cout << "You entered: " << userInput << std::endl;

    return 0;
}
```

Collection Framework Questions

6. **What is the difference between Array and Linked List?**

- **Array**: Fixed size, contiguous memory allocation, direct access via index (O(1)), insertion/deletion requires shifting elements (O(n))

- **Linked List**: Dynamic size, non-contiguous memory, sequential access (O(n)), efficient insertion/deletion (O(1) at known position)
- Arrays are better for random access, while linked lists are better for frequent insertions/deletions

7. **What is ArrayIndexOutOfBoundsException?**

   - This exception occurs when trying to access an array element with an invalid index (negative or beyond array length)
   - Example: If array length is 5, accessing array[5] or array[-1] will throw ArrayIndexOutOfBoundsException
   - It's a runtime exception that indicates a programming error
   - Can be prevented by checking array bounds before accessing elements

SQL Questions

8. **Write a simple query to print max salary and the employee name**

```
SELECT employee_name, salary
FROM employees
WHERE salary = (SELECT MAX(salary) FROM employees);
```

9. **What is the difference between DELETE and TRUNCATE?**

   - **DELETE**:
     - DML command that removes specific rows based on a WHERE condition
     - Can be rolled back (transaction-safe)
     - Triggers are fired
     - Slower than TRUNCATE as it logs individual row deletions
   - **TRUNCATE**:
     - DDL command that removes all rows from a table
     - Cannot be rolled back (commits automatically)
     - Doesn't fire triggers
     - Faster as it deallocates pages rather than rows
     - Resets identity columns (if any)

10. **Find employees with similar names**

```
SELECT name, COUNT(*)
FROM employees
GROUP BY name
HAVING COUNT(*) > 1;
```

11. **Find the 3rd least time working employee**

```
-- Using LIMIT with offset
SELECT employee_name, time_worked
FROM employees
ORDER BY time_worked ASC
LIMIT 1 OFFSET 2;
```

12. **Normalization**

    - Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity
    - **First Normal Form (1NF)**: Eliminate duplicate columns, create separate tables for related data, identify each row with a unique column or set of columns (primary key)
    - **Second Normal Form (2NF)**: Meet 1NF requirements and remove subsets of data that apply to multiple rows to separate tables, create relationships via foreign keys
    - **Third Normal Form (3NF)**: Meet 2NF requirements and remove columns not dependent on the primary key
    - **BCNF (Boyce-Codd Normal Form)**: More stringent version of 3NF
    - **Fourth Normal Form (4NF)**: Addresses multi-valued dependencies
    - **Fifth Normal Form (5NF)**: Addresses join dependencies

13. **Triggers**

    - Triggers are database objects that automatically execute when a specified database event occurs
    - Events can include INSERT, UPDATE, DELETE operations
    - Types of triggers: BEFORE, AFTER, INSTEAD OF
    - Uses include: enforcing business rules, maintaining data integrity, auditing changes, implementing complex security
    - Example:

      ```
      CREATE TRIGGER update_last_modified
      BEFORE UPDATE ON employees
      FOR EACH ROW
      SET NEW.last_modified = NOW();
      ```

14. **Acid properties**

    - **Atomicity**: Transactions are all-or-nothing (either fully completed or fully rolled back)
    - **Consistency**: Transactions bring the database from one valid state to another (constraints, cascades, triggers)
    - **Isolation**: Concurrent transactions should not affect each other (transaction isolation levels control this)
    - **Durability**: Once a transaction is committed, it remains committed even in case of system failure (changes are permanent)

15. **Joins**

    - **INNER JOIN**: Returns records with matching values in both tables
    - **LEFT JOIN**: Returns all records from the left table and matched records from the right table
    - **RIGHT JOIN**: Returns all records from the right table and matched records from the left table
    - **FULL JOIN**: Returns all records when there is a match in either left or right table
    - **CROSS JOIN**: Returns the Cartesian product of both tables
    - **SELF JOIN**: Joining a table to itself

16. **Aggregate function**

    - Functions that perform calculations on multiple rows and return a single value

- Common aggregate functions: COUNT(), SUM(), AVG(), MIN(), MAX()
- Usually used with GROUP BY clause
- Example:

```
SELECT department, AVG(salary) as avg_salary
FROM employees
GROUP BY department;
```

Object-Oriented Programming Questions

17. **What are the main features of OOP?**

- **Encapsulation**: Bundling data and methods together, restricting direct access to data
- **Inheritance**: Ability to create new classes from existing ones
- **Polymorphism**: Ability of objects to take on many forms depending on context
- **Abstraction**: Hiding implementation details and showing only functionality

18. **What is abstraction?**

- Abstraction is hiding implementation details and showing only essential features
- It helps manage complexity by focusing on what an object does rather than how it does it
- In Java, abstraction is achieved using abstract classes and interfaces
- Example: Car interface with methods like start(), stop(), accelerate() without implementation details

19. **Difference between abstraction and encapsulation**

- **Abstraction**: Focuses on what an object does (interface), hides complex implementation details
- **Encapsulation**: Focuses on how to achieve functionality, binds data and behavior together, controls access to data
- Abstraction operates at design level, encapsulation at implementation level
- Abstraction is implemented using interfaces and abstract classes, encapsulation using access modifiers
- Example: An abstract class Car defines what a car should do, while encapsulation ensures engine details are hidden from the user

20. **Polymorphism and its types**

- Polymorphism means "many forms" - ability of an object to take on many forms
- **Compile-Time/Static Polymorphism**:
  - Method overloading (same method name, different parameters)
  - Operator overloading (in languages that support it)
  - Resolved during compilation
- **Runtime/Dynamic Polymorphism**:
  - Method overriding (subclass implements method of parent class)
  - Uses inheritance and interfaces
  - Resolved during runtime
- Example: Shape class with draw() method, subclasses Circle, Rectangle override the method

Data Manipulation

21. **How can I show the employee names who have similar names?**

```
-- SQL approach for exact duplicates
SELECT name, COUNT(*)
FROM employees
GROUP BY name
HAVING COUNT(*) > 1;
```

Project and Scenario-Based Questions

22. **Scenario-based question: Collecting student details (roll no, name, mobile)**
    - Design considerations:
        - Data validation (roll numbers unique, phone numbers valid)
        - Storage (database schema design)
        - User interface for input and retrieval
    - Database design:

```
CREATE TABLE Students (
    roll_number VARCHAR(20) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    mobile VARCHAR(15) CHECK (mobile ~ '^[0-9]+$'),
    registration_date DATE DEFAULT CURRENT_DATE
);
```

    - API endpoints needed:
        - POST /students - Add new student
        - GET /students - Retrieve all students
        - GET /students/ - Retrieve specific student
        - PUT /students/ - Update student details
        - DELETE /students/ - Remove student

JavaScript Questions

(generic, as specific questions weren't detailed)

23. **JavaScript related questions**
    - **What is the difference between let, const, and var?**
        - `var`: Function-scoped, hoisted, can be redeclared
        - `let`: Block-scoped, not hoisted, cannot be redeclared in same scope
        - `const`: Block-scoped, not hoisted, cannot be reassigned after declaration
    - **What are closures in JavaScript?**
        - Closures are functions that have access to variables from an outer function's scope even after the outer function has finished executing
    - **How does event delegation work?**
        - Technique of adding event listeners to parent elements rather than individual children
        - Utilizes event bubbling to handle events at a higher level in the DOM

Cloud Computing

24. **Cloud computing knowledge Well Explained (https://youtu.be/61Ts2KAdxyY?si=owuhVj_7G3E9QOLm)**
    - **Cloud Computing**: Delivery of computing services over the internet
    - **Service Models**:
        - Infrastructure as a Service (IaaS): Provides virtualized computing resources
        - Platform as a Service (PaaS): Provides platform for application development
        - Software as a Service (SaaS): Delivers software applications over the internet
    - **Deployment Models**: Public, Private, Hybrid, Community
    - **Key Benefits**: Scalability, cost-efficiency, flexibility, disaster recovery
    - **Major Providers**: AWS, Microsoft Azure, Google Cloud Platform
    - **Key Technologies**: Virtualization, containerization, serverless computing

Personal Questions

25. **Self introduction**

    - Keep it professional but personable
    - Structure: Brief background, education, relevant experience, technical skills, career goals
    - Focus on aspects relevant to the position
    - Keep it concise (1-2 minutes)

26. **What is the most disliked thing in college and how did you overcome it?**

    - Possible answer: "The rigid structure of some courses limited creative problem-solving. I overcame this by joining programming clubs and working on personal projects that allowed me to apply classroom concepts in more flexible, real-world contexts."
    - Show resilience and problem-solving ability
    - Demonstrate personal growth
    - Keep it honest but positive

Machine Learning

27. **Project-related machine learning questions (moderate level)**
    - **Supervised vs. Unsupervised Learning**:
        - Supervised: Training with labeled data
        - Unsupervised: Finding patterns in unlabeled data
    - **Common Algorithms**:
        - Classification: Decision Trees, Random Forest, SVM, Neural Networks
        - Regression: Linear Regression, Ridge, Lasso
        - Clustering: K-means, Hierarchical, DBSCAN
    - **Evaluation Metrics**:
        - Classification: Accuracy, Precision, Recall, F1-score, ROC-AUC
        - Regression: MSE, RMSE, MAE, R^2
    - **Overfitting and Underfitting**:
        - Causes, detection, and remedies (regularization, cross-validation)
    - **Feature Selection/Engineering**: Importance and techniques

Deep Learning

28. **Deep learning questions (moderate level)**
    - **Neural Network Basics**: Layers, neurons, activation functions
    - **Common Architectures**: CNN, RNN, LSTM, Transformers
    - **Training Process**: Backpropagation, gradient descent
    - **Frameworks**: TensorFlow, PyTorch, Keras
    - **Applications**: Computer vision, NLP, speech recognition
    - **Transfer Learning**: Using pre-trained models

Additional Topics

29. **Topic discussion (PIG)**
    - **Pig Latin**: Animal.

MASSIVE SUCCESS RATE

"Transform Your Interview Opportunity into an Offer Letter and Make Your Parents Proud!"

- **In-depth Technical Mock**
  - **Crack coding challenges with real experts.**
- **HR & Managerial Prep**
  - **Master behavioral questions and impress TCS.**
- **Full Interview Simulation**
  - **Ace both technical and HR in one session.**
- **Resume Review**
  - **Identify and fix weaknesses for a standout CV.**
- **Personalized Feedback & Expert Guidance**
  - **Tailored improvement tips to boost success.**

www.primecoding.in