# Software Developer Technical Assessment

**Duration:** 2 hours

**Instructions:**

- Complete as many questions as possible within the allotted time.
- There are 6 sections and 1 Bonus section. You can do them in any order, the recommended time for each section is also listed.
- Questions vary in difficulty and are weighted accordingly.
- When answering questions with code, You may use standard libraries of your preferred programming language, but no external frameworks or libraries unless specified.

---

## Section 1: Programming and Problem Solving (20 minutes)

### Question 1.1: Array Manipulation (15 points)

Write a function that takes an array of integers and returns the pair of integers whose sum is closest to zero. If there are multiple pairs with the same closest sum, return any one of them.

Example:

Input: [1, 4, -3, -1, 5, 9]
Output: [1, -1]

### Question 1.2: String Processing (15 points)

Write a function that checks if two strings are anagrams of each other, ignoring spaces and case sensitivity.

Example:

Input: "Listen", "Silent"
Output: true

Input: "Hello", "World"
Output: false

---

## Section 2: Object-Oriented Programming (15 minutes)

### Question 2.1: Class Design (25 points)

Design a simple Library Management System using OOP principles. Your system should include the following classes:

- Book
- Author
- Library
- User

The system aims to have functionality for:

- Adding books to the library
- Checking out books
- Returning books
- Searching for books by title or author

Draw a class diagram to capture the relationships, write function signatures that would be present for each class. (you do not need to write the whole logic)

Ensure proper encapsulation, inheritance (where appropriate), and demonstrate polymorphism.

### Question 2.2: Design Patterns (20 points)

Explain the Singleton design pattern. Discuss one scenario where using a Singleton would be appropriate and one scenario where it should be avoided.

---

## Section 3: Database (20 minutes)

### Question 3.1: SQL Queries (25 points)

Given the following database schema:

**Employees**

id (INT, Primary Key)
name (VARCHAR)
department_id (INT, Foreign Key)

salary (DECIMAL)
hire_date (DATE)
manager_id (INT, Foreign Key referencing id)

**Departments**

id (INT, Primary Key)
name (VARCHAR)
location (VARCHAR)
budget (DECIMAL)

**Projects**

id (INT, Primary Key)
name (VARCHAR)
start_date (DATE)
end_date (DATE)
department_id (INT, Foreign Key)

**EmployeeProjects**

employee_id (INT, Foreign Key)
project_id (INT, Foreign Key)
hours_worked (INT)
role (VARCHAR)

Write SQL queries for:

a) Find all employees in the "Engineering" department sorted by their salary in descending order.

b) List all projects along with the total number of employees assigned to each project.

c) Find the department with the highest average salary.

## Question 3.2: Database Design (20 points)

Design a database schema for an e-commerce platform with the following requirements:

- Users can create accounts and place orders
- Products belong to multiple categories
- Orders contain multiple products with quantities

- Users can leave reviews for products they've purchased
- Products have inventory tracking

Draw the ER diagram and write the CREATE TABLE statements with proper constraints.

---

## Section 4: Networking and Web (20 minutes)

### Question 4.1: HTTP and REST (15 points)

Explain the following concepts:

- Difference between HTTP methods GET, POST, PUT, and DELETE
- RESTful API design principles
- HTTP status codes and their meanings
- Web cookies and their purpose

### Question 4.2: Application Layer Protocols (15 points)

Answer the following questions briefly:

a) What are the differences between TCP and UDP protocols. When would you use one over the other?

b) Describe the process of DNS resolution from typing a URL in a browser to loading the webpage.

c) What is HTTPS? How does it differ from HTTP in terms of security?

---

## Section 5: Software Engineering Principles (15 minutes)

### Question 5.1: Testing (15 points)

Write unit tests for the following function using any testing framework of your choice:

```
function calculateDiscount(purchaseAmount, membershipLevel) {
  if (membershipLevel === 'gold') {
    return purchaseAmount * 0.15;
  } else if (membershipLevel === 'silver') {
    return purchaseAmount * 0.10;
  } else if (membershipLevel === 'bronze') {
    return purchaseAmount * 0.05;
  } else {
    return 0;
  }
}
```

Write test cases that cover all possible scenarios and edge cases. You can write pseudo code.

### Question 5.2: Software Development Life Cycle (20 points)

a) What is a Git branch and how would you use branching in a team development environment?

b) What are some DevOps practices that can improve the software development process?

---

## Section 6: Practical Implementation (30 minutes)

### Question 6.1: Full Stack Implementation (30 points)

Implement a simple task management application with the following features:

- Display a list of tasks (title, description, due date, status)
- Add a new task
- Mark a task as complete
- Delete a task

You can choose any of the following technology stacks:

1. Frontend: HTML, CSS, JavaScript (vanilla or with a framework) Backend: RESTful API using language of your choice

2. Console-based application in your language of choice with file-based storage

**Data for tasks.json:**

```
[
  {
    "id": 1,
    "title": "Complete project proposal",
    "description": "Draft the proposal for the new client project",
    "dueDate": "2025-05-10",
    "completed": false
```

```json
  },
  {
    "id": 2,
    "title": "Review pull requests",
    "description": "Review and merge team pull requests",
    "dueDate": "2025-04-30",
    "completed": true
  },
  {
    "id": 3,
    "title": "Update documentation",
    "description": "Update API documentation with new endpoints",
    "dueDate": "2025-05-05",
    "completed": false
  },
  {
    "id": 4,
    "title": "Fix login bug",
    "description": "Address the authentication issue reported by QA",
    "dueDate": "2025-04-29",
    "completed": false
  }
]
```

Focus on code quality, organization, and proper implementation of software design principles.

---

## Bonus Question (Optional)

### Memory Management (10 bonus points)

Explain the concept of memory leaks, how they occur in different programming languages, and strategies to prevent them. Provide a small code example in your preferred language that demonstrates a potential memory leak and how to fix it.

---

**Good luck!**