SET 1

Q1. (Data Structures – 5 Marks)

Using AI assistance, write a Python program to implement a **Stack** using a list.
Perform the following operations:

- Push 5 elements
- Pop 2 elements
- Display the remaining stack
  Explain how AI helped generate or optimize your structure.

Q2. (Web Frontend – 5 Marks)

With AI tools, develop an interactive **To-Do List App** using HTML, CSS, and JavaScript.
Include features:

- Add task
- Delete task
- Mark as completed

==========

SET 2

Q1. (Algorithms – 5 Marks)

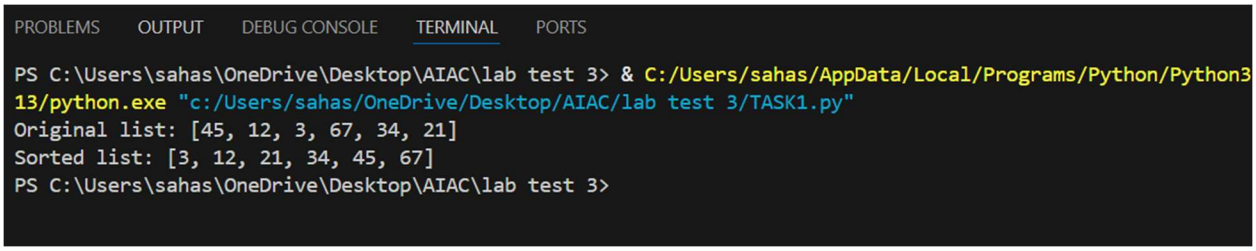
Using AI assistance, generate Python code to implement **Merge Sort**.
Run the algorithm on the list: `45, 12, 3, 67, 34, 21`.
Explain the time complexity and how AI improved correctness.

CODE:

```python
def merge_sort(arr):
    """
    Sort an array using the Merge Sort algorithm.
    Time Complexity: O(n log n) - all cases
    Space Complexity: O(n)
    """
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left = merge_sort(arr[:mid])
    right = merge_sort(arr[mid:])

    return merge(left, right)


def merge(left, right):
    """Merge two sorted arrays into one sorted array."""
    result = []
    i = j = 0

    while i < len(left) and j < len(right):
        if left[i] <= right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            i += 1
```

```python
        result.extend(left[i:])
        result.extend(right[j:])

        return result


    # Test with the given list
    arr = [45, 12, 3, 67, 34, 21]
    print(f"Original list: {arr}")
    sorted_arr = merge_sort(arr)
    print(f"Sorted list: {sorted_arr}")
```

OUTPUT:



Q2. (Web Frontend – 5 Marks)

Use AI to generate a webpage showing a **student registration form** with:

- Input fields (Name, Email, Course)
- Submit button
- JavaScript form validation

CODE:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Registration Form</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            max-width: 500px;
            margin: 50px auto;
            padding: 20px;
            background-color: #f4f4f4;
        }
        .form-container {
            background-color: white;
            padding: 30px;
            border-radius: 8px;
            box-shadow: 0 2px 10px rgba(0,0,0,0.1);
        }
        h1 {
            text-align: center;
            color: #333;
        }
        .form-group {
            margin-bottom: 20px;
        }
        label {
            display: block;
```

```css
label {
    display: block;
    margin-bottom: 5px;
    color: #555;
    font-weight: bold;
}
input, select {
    width: 100%;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 4px;
    box-sizing: border-box;
    font-size: 14px;
}
input:focus, select:focus {
    outline: none;
    border-color: #4CAF50;
    box-shadow: 0 0 5px rgba(76,175,80,0.3);
}
button {
    width: 100%;
    padding: 12px;
    background-color: #4CAF50;
    color: white;
    border: none;
    border-radius: 4px;
```

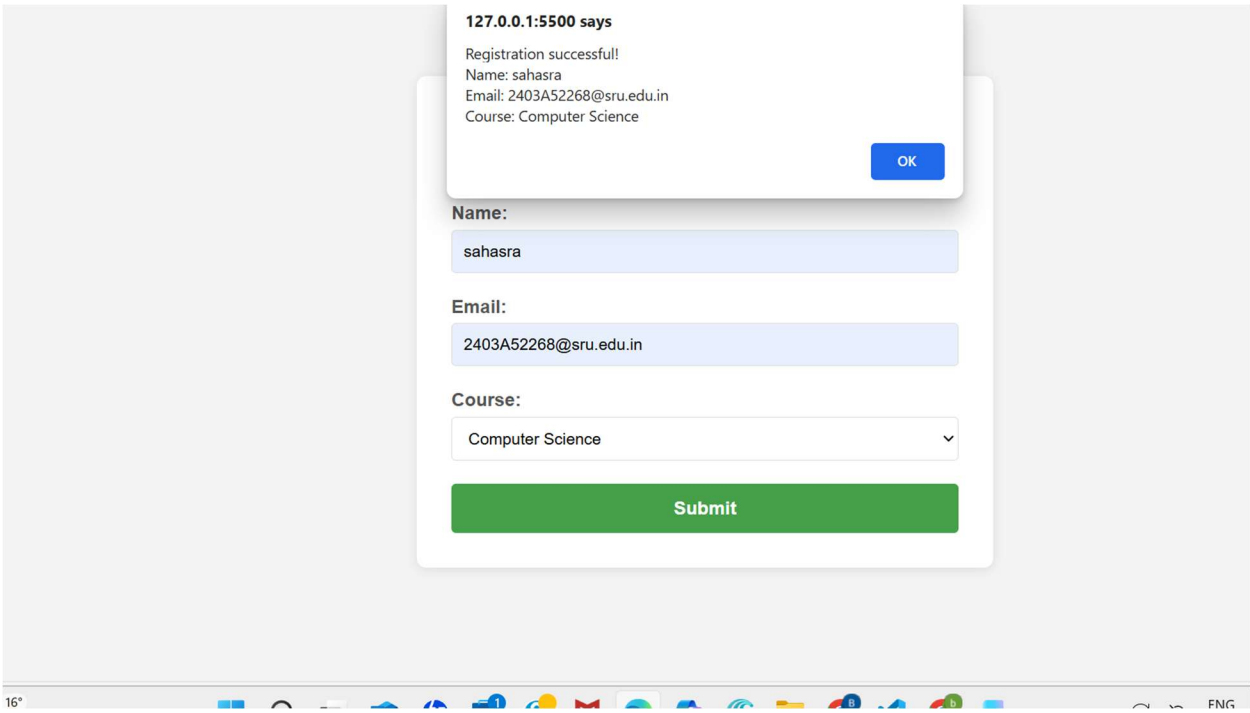
```
        font-size: 16px;
        cursor: pointer;
        font-weight: bold;
    }
    button:hover {
        background-color: #45a049;
    }
    .error {
        color: #d32f2f;
        font-size: 12px;
        margin-top: 5px;
    }
</style>
ead>
dy>
<div class="form-container">
    <h1>Student Registration Form</h1>
    <form id="registrationForm">
        <div class="form-group">
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required>
            <div class="error" id="nameError"></div>
        </div>

        <div class="form-group">
```

```
            <div class="form-group">
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" required>
                <div class="error" id="emailError"></div>
            </div>

            <div class="form-group">
                <label for="course">Course:</label>
                <select id="course" name="course" required>
                    <option value="">Select a Course</option>
                    <option value="Computer Science">Computer Science</option>
                    <option value="Engineering">Engineering</option>
                    <option value="Business">Business</option>
                    <option value="Arts">Arts</option>
                </select>
                <div class="error" id="courseError"></div>
            </div>

            <button type="submit">Submit</button>
        </form>
    </div>

    <script>
        document.getElementById('registrationForm').addEventListener('submit', function(e) {
            e.preventDefault();
```

```
    document.getElementById('registrationForm').addEventListener('submit', function(e) {
        document.getElementById('nameError').textContent = '';
        document.getElementById('emailError').textContent = '';
        document.getElementById('courseError').textContent = '';

        const name = document.getElementById('name').value.trim();
        const email = document.getElementById('email').value.trim();
        const course = document.getElementById('course').value;

        let isValid = true;

        // Validate Name
        if (name === '') {
            document.getElementById('nameError').textContent = 'Name is required';
            isValid = false;
        } else if (name.length < 3) {
            document.getElementById('nameError').textContent = 'Name must be at least 3 character
            isValid = false;
        }

        // Validate Email
        const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        if (email === '') {
            document.getElementById('emailError').textContent = 'Email is required';
            isValid = false;
        } else if (!emailPattern.test(email)) {
```

```
            // Validate Email
            const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
            if (email === '') {
                document.getElementById('emailError').textContent = 'Email is required';
                isValid = false;
            } else if (!emailPattern.test(email)) {
                document.getElementById('emailError').textContent = 'Please enter a valid email';
                isValid = false;
            }

            // Validate Course
            if (course === '') {
                document.getElementById('courseError').textContent = 'Please select a course';
                isValid = false;
            }

            if (isValid) {
                alert('Registration successful!\nName: ' + name + '\nEmail: ' + email + '\nCourse: '
                document.getElementById('registrationForm').reset();
            }
        });
    </script>
</body>
</html>
```

OUTPUT



==========

SET 3

Q1. (Data Structures – 5 Marks)

Use AI to create a Python program implementing a **Queue** using the `collections.deque` module.
Perform:

- Enqueue 4 values
- Dequeue 1 value
- Display queue
  Explain AI's suggestion accuracy.

Q2. (Algorithms – 5 Marks)

With AI assistance, implement **Binary Search** in Python.
Test it on the sorted array:
`[10, 20, 30, 40, 50, 60]` to search for the number **40**.

==========

SET 4

Q1. (Algorithms – 5 Marks)

Use AI to generate Python code for **Bubble Sort** and **Insertion Sort**.
Compare their execution time using Python's `time` module.

Q2. (Web Frontend – 5 Marks)

With AI help, create a webpage showing a **dynamic product list** using JavaScript arrays.
Each product must display:

- Name
- Price
- "Add to Cart" button

==========

SET 5

Q1. (Data Structures – 5 Marks)

Using AI tools, implement a **Linked List** in Python with operations:

- Insert at beginning
- Insert at end
- Delete a node
  Display the list after operations.

Q2. (Web Frontend – 5 Marks)

With AI assistance, generate a responsive **Portfolio Website Layout** including:

- Header
- About section
- Projects section
- Contact form