Q1. Reverse a Stack You are given a stack of integers. Your task is to reverse the order of the elements in the stack using only stack operations (push and pop) and without using any additional data structures. Ex. stack = [1, 2, 3, 4, 5] reverse Stack(stack) print(stack) Output should be [5, 4, 3, 2, 1] ?.

❏**Source code**

```
def insert_at_bottom(stack, element):
    if not stack:
        stack.append(element)
    else:
        top = stack.pop()
        insert_at_bottom(stack, element)
        stack.append(top)
def reverse_stack(stack):
    if stack:
        top = stack.pop()
        reverse_stack(stack)
        insert_at_bottom(stack, top)
stack = [1, 2, 3, 4, 5]
reverse_stack(stack)
print(stack)
```

❏**Output:-**
[5, 4, 3, 2, 1]

Q2. Three Sum Problem Sample Problem: Given an array of integers, find all unique triplets in the array which give the sum of zero. The solution should return the list of triplets.?

❑Source code:

```python
def three_sum(nums):
    nums.sort()
    result = []
    for i in range(len(nums) - 2):
        if i > 0 and nums[i] == nums[i - 1]:
            continue
        left, right = i + 1, len(nums) - 1
        while left < right:
            total = nums[i] + nums[left] + nums[right]
            if total == 0:
                result.append([nums[i], nums[left], nums[right]])
                while left < right and nums[left] == nums[left + 1]:
                    left += 1
                while left < right and nums[right] == nums[right - 1]:
                    right -= 1
                left += 1
                right -= 1
            elif total < 0:
                left += 1
            else:
                right -= 1
    return result

# Example usage nums = [-1, 0, 1, 2, -1, -4]
print(three_sum(nums))
```
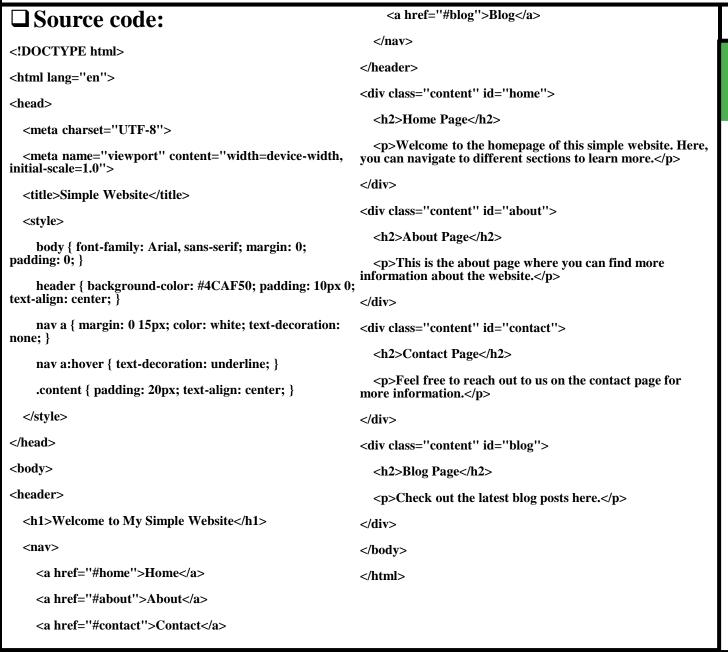
Output:
[[-1, -1, 2], [-1, 0, 1]]

Q3. Depth-First Search (DFS) Sample Problem: Implement Depth First Search (DFS) to traverse a graph starting from a given vertex. The graph is represented by an adjacency list

□Source code:

```python
# Function to perform DFS

def dfs(graph, start, visited=None):
    if visited is None:
        visited = set()  # Initialize visited set if not provided
    visited.add(start)
    print(start, end=' ')  # Print the visited node
    for neighbor in graph[start]:
        if neighbor not in visited:
            dfs(graph, neighbor, visited)
```

```python
# Example usage

graph = {
    0: [1, 2],
    1: [0, 3, 4],
    2: [0, 5],
    3: [1],
    4: [1],
    5: [2]
}
# Start DFS from vertex 0
dfs(graph, 0)
```

Output:
0 1 3 4 2 5

Q4. Create a Simple Website with the Following Features: a. Display a welcoming message and a brief description. b. Include navigation links to the homepage, about page, contact page, and blog page ?.

❑ Source code:

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Simple Website</title>
  <style>
    body { font-family: Arial, sans-serif; margin: 0; padding: 0; }
    header { background-color: #4CAF50; padding: 10px 0; text-align: center; }
    nav a { margin: 0 15px; color: white; text-decoration: none; }
    nav a:hover { text-decoration: underline; }
    .content { padding: 20px; text-align: center; }
  </style>
</head>
<body>
<header>
  <h1>Welcome to My Simple Website</h1>
  <nav>
    <a href="#home">Home</a>
    <a href="#about">About</a>
    <a href="#contact">Contact</a>
    <a href="#blog">Blog</a>
  </nav>
</header>
<div class="content" id="home">
  <h2>Home Page</h2>
  <p>Welcome to the homepage of this simple website. Here, you can navigate to different sections to learn more.</p>
</div>
<div class="content" id="about">
  <h2>About Page</h2>
  <p>This is the about page where you can find more information about the website.</p>
</div>
<div class="content" id="contact">
  <h2>Contact Page</h2>
  <p>Feel free to reach out to us on the contact page for more information.</p>
</div>
<div class="content" id="blog">
  <h2>Blog Page</h2>
  <p>Check out the latest blog posts here.</p>
</div>
</body>
</html>
```

Output: