LAB TASK(week-4)

Name: Ahmad Raza

Sap I’d: 54471

Question #1

```
#include <iostream>
using namespace std;

class Stack {
private:
   int* data;       // Pointer to the stack's array
   int capacity;    // Maximum number of elements the stack can
hold     int top;        // Index of the top element in the stack

public:
   // Constructor
   Stack(int ignored = 0) : capacity(100), top(-1) {
      data = new int[capacity]; // Initialize stack with default capacity
   }

   // Destructor
~Stack() {
      delete[] data; // Deallocate memory used by the stack
```

```
    }

    // Push element onto the stack
void push(int item) {
        if (top + 1 >= capacity) { // Check if stack is full
cout << "Stack is full. Cannot push " << item << endl;
return;
        }
        data[++top] = item;
    }

    // Pop element from the
stack     void pop() {         if
(isEmpty()) {
            cout << "Stack is empty. Cannot pop." << endl;
return;
        }
        --top;
    }

    // Peek at the top element of the
stack     int peek() const {         if
(isEmpty()) {
```

```
        cout << "Stack is empty. Cannot peek." << endl;          return
-1; // Return an invalid value to indicate an error
    }
    return data[top];
  }

  // Clear all elements from the stack
void clear() {
    top = -1; // Reset top index to indicate an empty stack
  }

  // Check if the stack is
empty   bool isEmpty() const
{     return top == -1;
  }
};

int main() {
  Stack stack;

  // Test stack operations
  cout << "Pushing 10, 20, 30 onto the stack." <<
endl;   stack.push(10);   stack.push(20);
stack.push(30);
```

```
    cout << "Peek at top element: " << stack.peek() << endl;

    cout << "Popping top element." << endl;
stack.pop();

    cout << "Peek at top element after pop: " << stack.peek() << endl;

    cout << "Clearing stack." << endl;
stack.clear();

    if (stack.isEmpty()) {
        cout << "Stack is empty." << endl;
    } else {
        cout << "Stack is not empty." << endl;
    }

    return 0;
}
```

Question #2

```
#include <iostream>
```

```
#include <stack>
#include <string>

using namespace std;

int main() {    // Input
string    string str;    cout
<< "Enter a string: ";
getline(cin, str);

   // Stack to store characters
stack<char> charStack;

   // Push each character of the string into the
stack    for (char ch : str) {       charStack.push(ch);
   }

   // Pop characters from the stack to reverse the
string    string reversedStr;    while
(!charStack.empty()) {       reversedStr +=
charStack.top();       charStack.pop();
   }

   // Output the reversed string
```

```
    cout << "Reversed string: " << reversedStr << endl;

    return 0;
}
```