

Presented by: **Maryam Siddiqui**

**www.linkedin.com/in/maryam-siddiqui-ghani**

Task: **Simple Calculator**

**Project Report: To-Do List Manager**

**Introduction:**

The "To-Do List Manager" project is a console application developed using C++ as part of an internship task for CodeSoft. The primary objective of this project is to create a simple and interactive to-do list manager that allows users to add, view, mark tasks as completed, and remove tasks from their to-do list. The application provides a user-friendly platform for organizing tasks efficiently.

**Project Scope:**

The scope of this project encompasses the development of a to-do list management application. The application is designed to offer users the ability to perform essential operations related to task management, such as adding tasks, viewing tasks, marking tasks as completed, and removing tasks from their to-do list.

**Features:**

**1. Interactive Console Interface:** The application provides an intuitive console interface that enables users to interact with the to-do list manager seamlessly.

**2. Adding Tasks:**

Users can add new tasks to their to-do list by entering task descriptions. Tasks are initialized as not completed by default.

**3. Viewing Tasks:**

The application displays the user's to-do list, including task numbers, completion status indicators, and task descriptions.

**4. Marking Tasks as Completed:**

Users can mark tasks as completed by selecting the corresponding task number. Completed tasks are visually identified.

**5. Removing Tasks:**

Users have the option to remove tasks from their to-do list by specifying the task number for removal.

**Implementation:**

The project is implemented in C++, employing a menu-driven approach to provide users with a range of task management functionalities. The core implementation steps include:

1. Displaying a welcoming title and message.

2. Utilizing a `do-while` loop to facilitate repeated menu-driven interactions.

3. Offering users a menu to choose from various task management operations.

4. Implementing functions for adding tasks, viewing tasks, marking tasks as completed and removing tasks.

5. Employing a vector of `Task` structures to store task information.

**Code Structure:**

The code structure is organized into distinct sections:

1. Displaying a welcoming title and message.

2. Employing a `do-while` loop for the primary menu-driven functionality.

3. Implementing functions for task management operations, including adding, viewing, marking, and removing tasks.

**Conclusion:**

The "To-Do List Manager" project displays the practical implementation of a console-based application that assists users in organizing their tasks effectively. The project underscores the use of fundamental programming concepts, including data structures, control structures, and interactive user interfaces. While the scope of the project is modest, it serves as a foundation for more comprehensive task management applications with expanded features and functionalities.

**Acknowledgments:**

This project was conceptualized and executed by Maryam Siddiqui during her internship at CodeSoft. Special acknowledgments go to mentors and guides for their invaluable support and guidance during the project's development.

**Contact Information:**

For inquiries or feedback regarding this project, kindly reach out to:

Maryam Siddiqui

\*Note: This project report provides a concise overview and can be expanded as per the specific requirements and details of the actual project.\*

**Source code:**

#include <iostream>

#include <vector>

#include <string>

using namespace std;

struct Task {

string description;

bool completed;

};

// Function prototypes

void displayMenu();

void addTask(vector<Task>& tasks);

void viewTasks(const vector<Task>& tasks);

void markCompleted(vector<Task>& tasks);

void removeTask(vector<Task>& tasks);

int main() {

vector<Task> tasks;

int choice;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Welcome to To-Do List Manager \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

do {

displayMenu();

cin >> choice;

switch (choice) {

case 1:

addTask(tasks);

break;

case 2:

viewTasks(tasks);

break;

case 3:

markCompleted(tasks);

break;

case 4:

removeTask(tasks);

break;

case 5:

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Thank you for using the To-Do List Manager! \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

break;

default:

cout << " Invalid choice. Please try again." << endl;

break;

}

} while (choice != 5);

return 0;

}

void displayMenu() {

// Display menu options

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Menu: \*" << endl;

cout << " \* 1. Add Task \*" << endl;

cout << " \* 2. View Tasks \*" << endl;

cout << " \* 3. Mark Task as Completed \*" << endl;

cout << " \* 4. Remove Task \*" << endl;

cout << " \* 5. Exit \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

cout << " Enter your choice: ";

}

void addTask(vector<Task>& tasks) {

Task newTask;

cout << " Enter task description: ";

cin.ignore();

getline(cin, newTask.description);

newTask.completed = false;

tasks.push\_back(newTask);

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Task added successfully! \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

}

void viewTasks(const vector<Task>& tasks) {

// Display the to-do list

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

cout << " To-Do List: " << endl;

cout << endl;

for (size\_t i = 0; i < tasks.size(); i++) {

cout << " " << (i + 1) << " . ";

if (tasks[i].completed)

cout << " [ X ] ";

else

cout << " [ ] ";

cout << tasks[i].description;

cout << string(50 - tasks[i].description.length(), ' ') << " " << endl;

}

cout << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

}

void markCompleted(vector<Task>& tasks) {

viewTasks(tasks);

int index;

cout << " Enter the number of the task to mark as completed: ";

cin >> index;

if (index >= 1 && index <= static\_cast<int>(tasks.size())) {

tasks[index - 1].completed = true;

// Task marked as completed confirmation

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Task marked as completed! \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

} else {

cout << " Invalid task number. Please try again." << endl;

}

}

void removeTask(vector<Task>& tasks) {

viewTasks(tasks);

int index;

cout << " Enter the number of the task to remove: ";

cin >> index;

if (index >= 1 && index <= static\_cast<int>(tasks.size())) {

tasks.erase(tasks.begin() + index - 1);

// Task removed confirmation

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " \* \*" << endl;

cout << " \* Task removed successfully! \*" << endl;

cout << " \* \*" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

} else {

cout << " Invalid task number. Please try again." << endl;

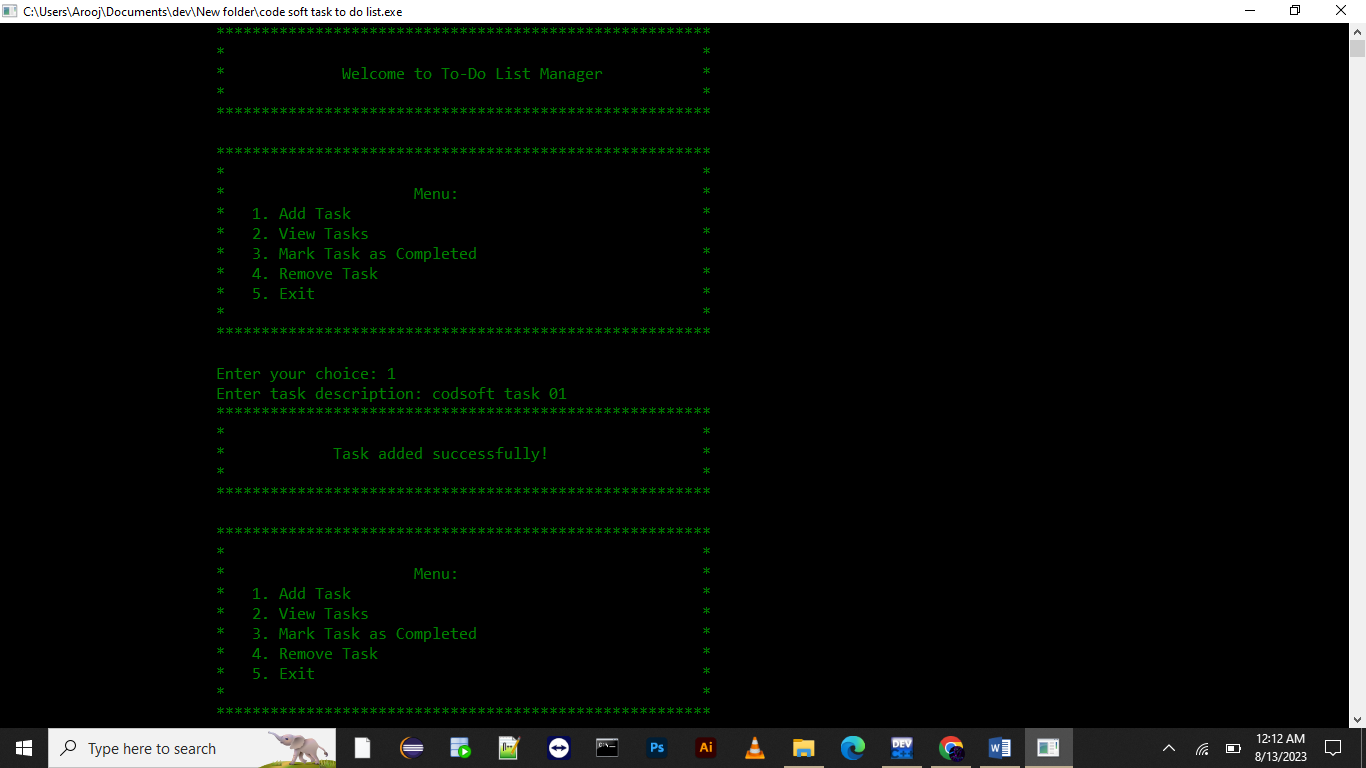
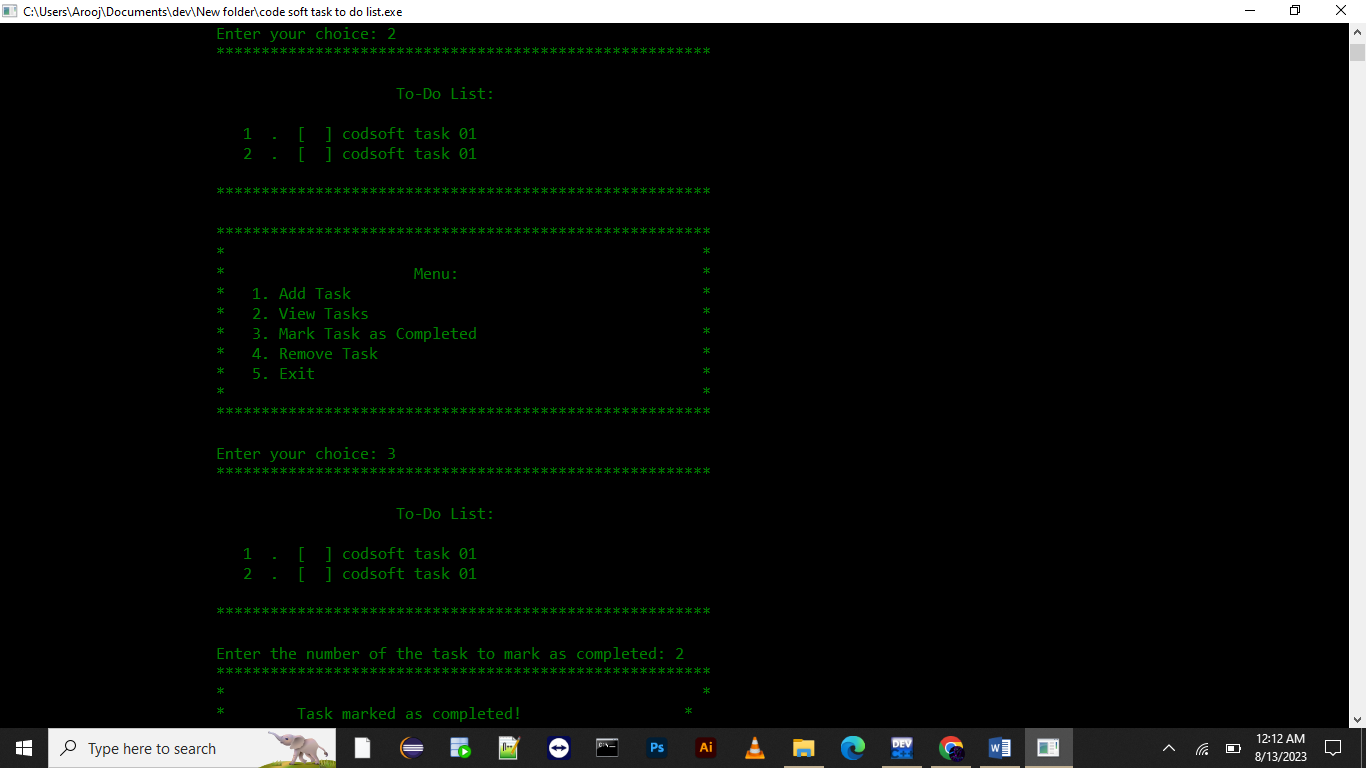
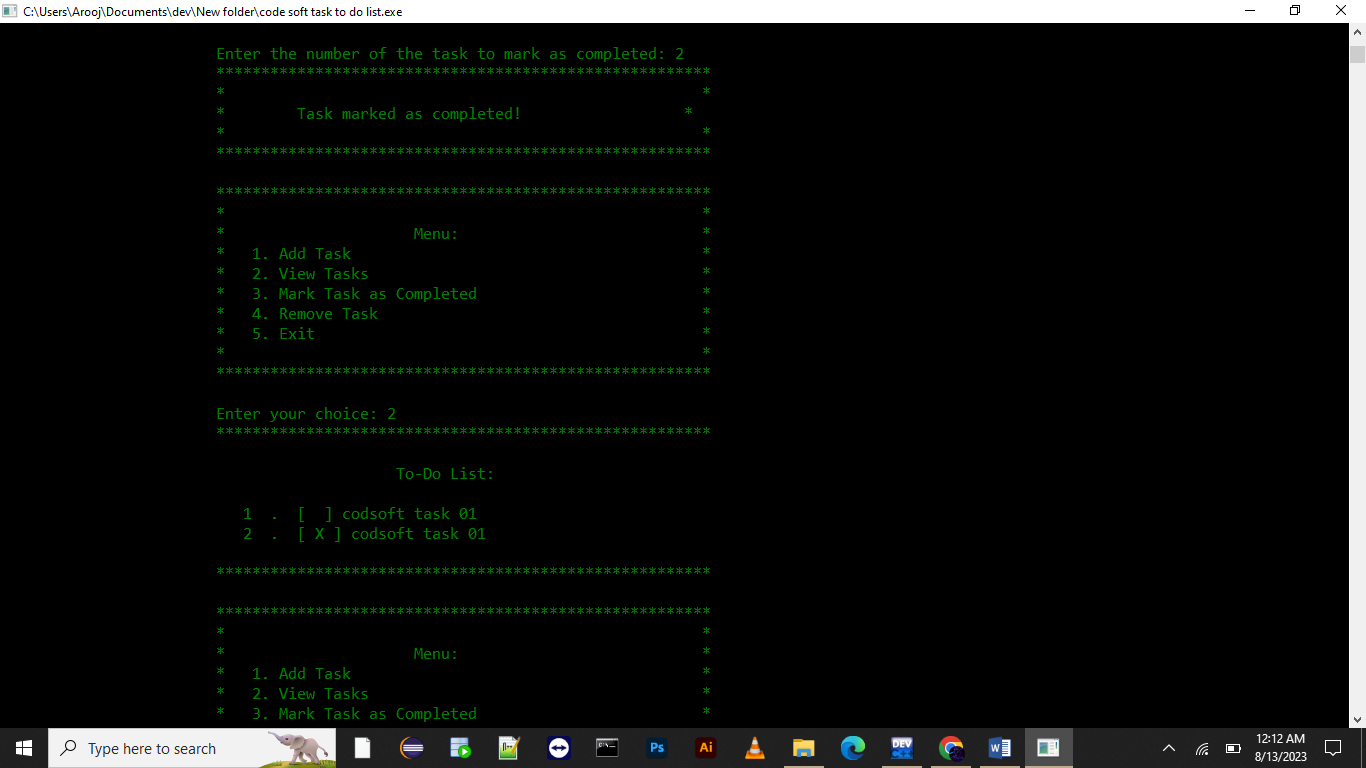
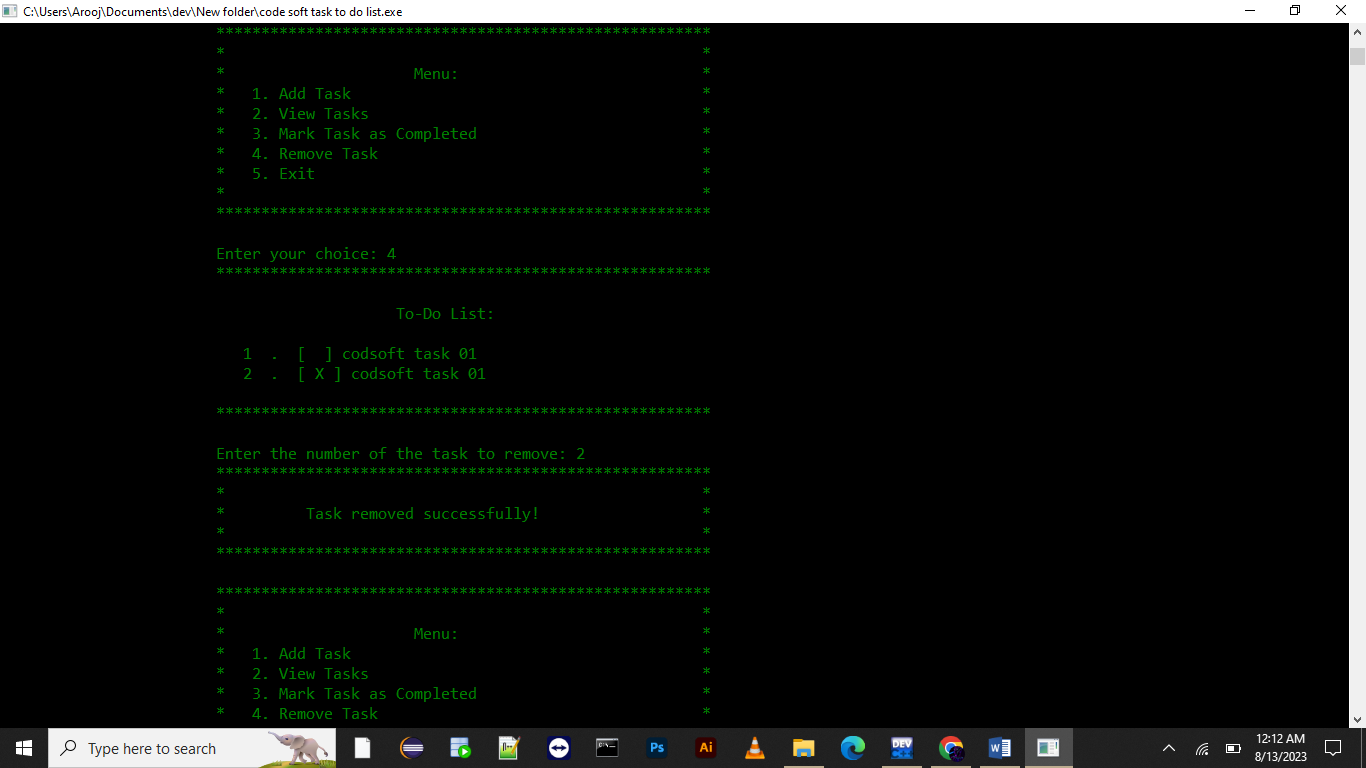
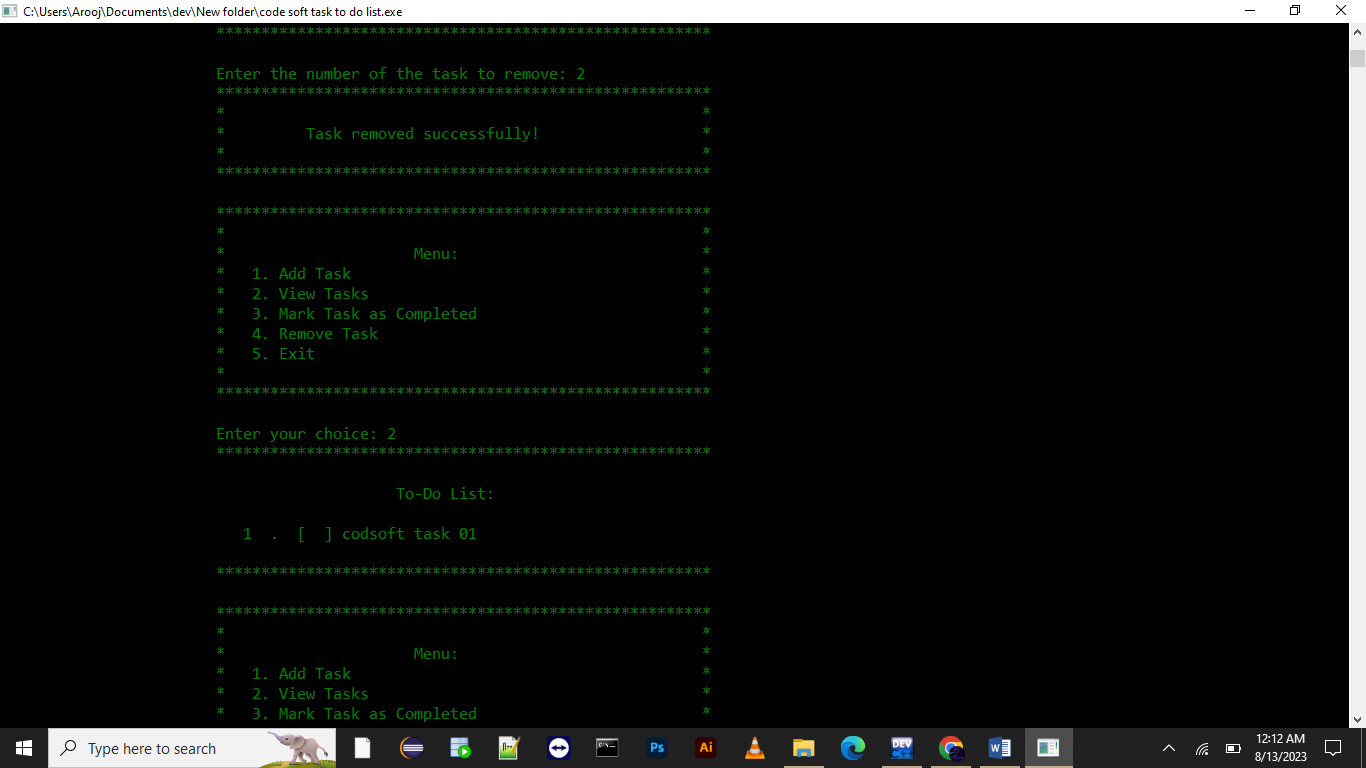
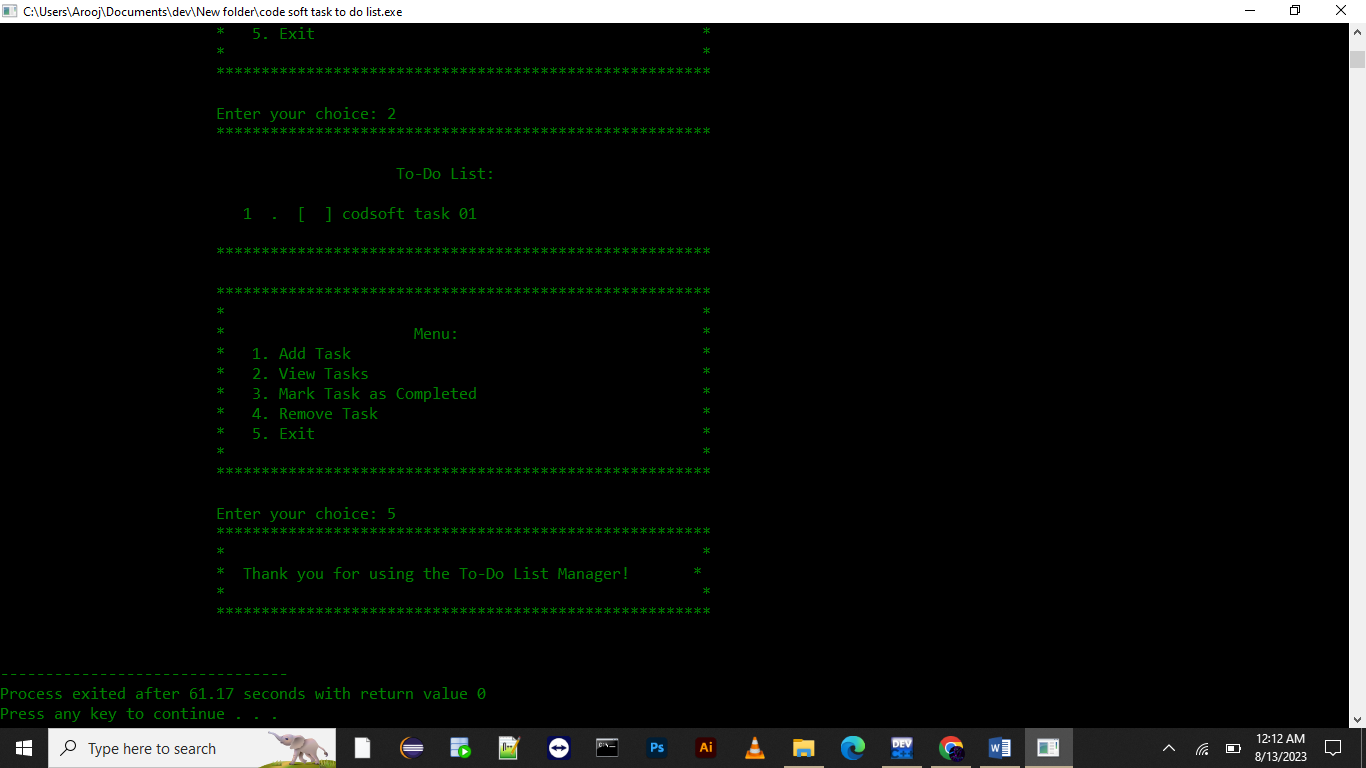
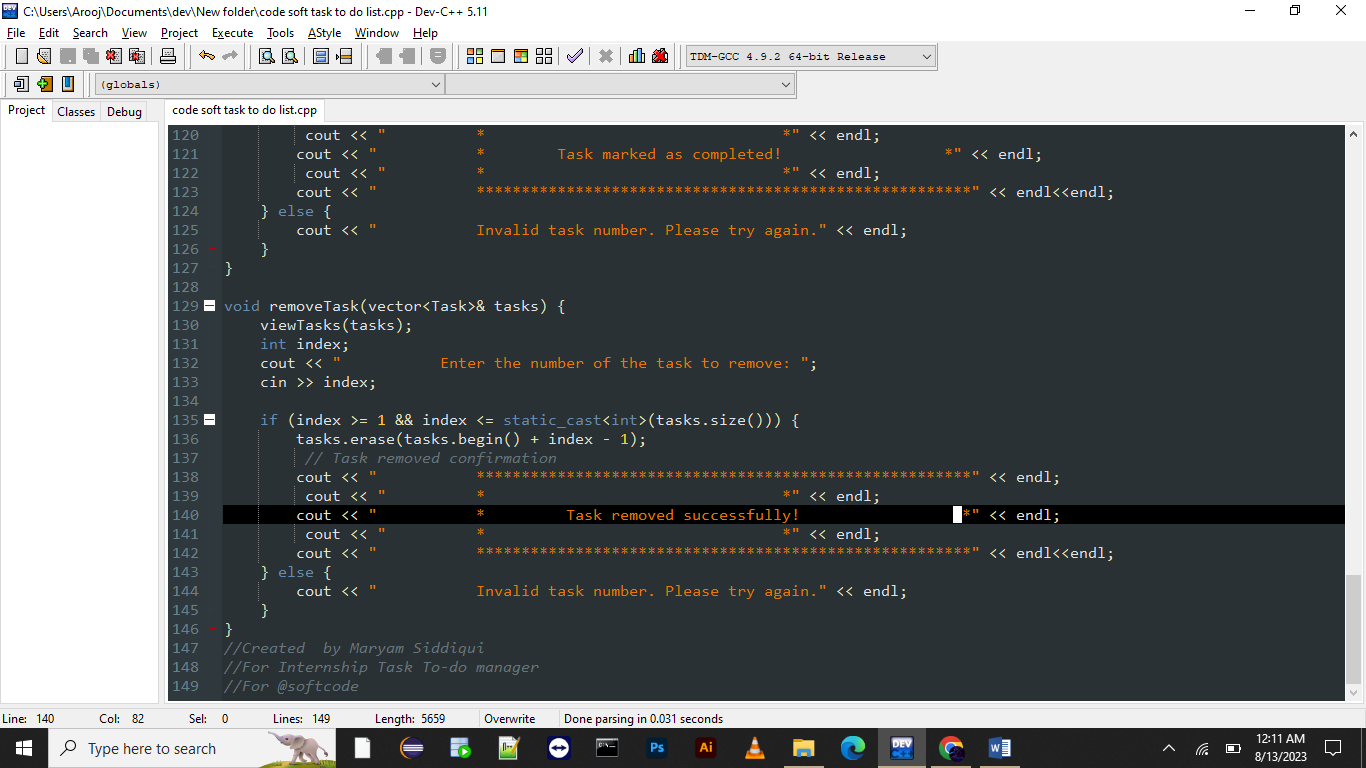
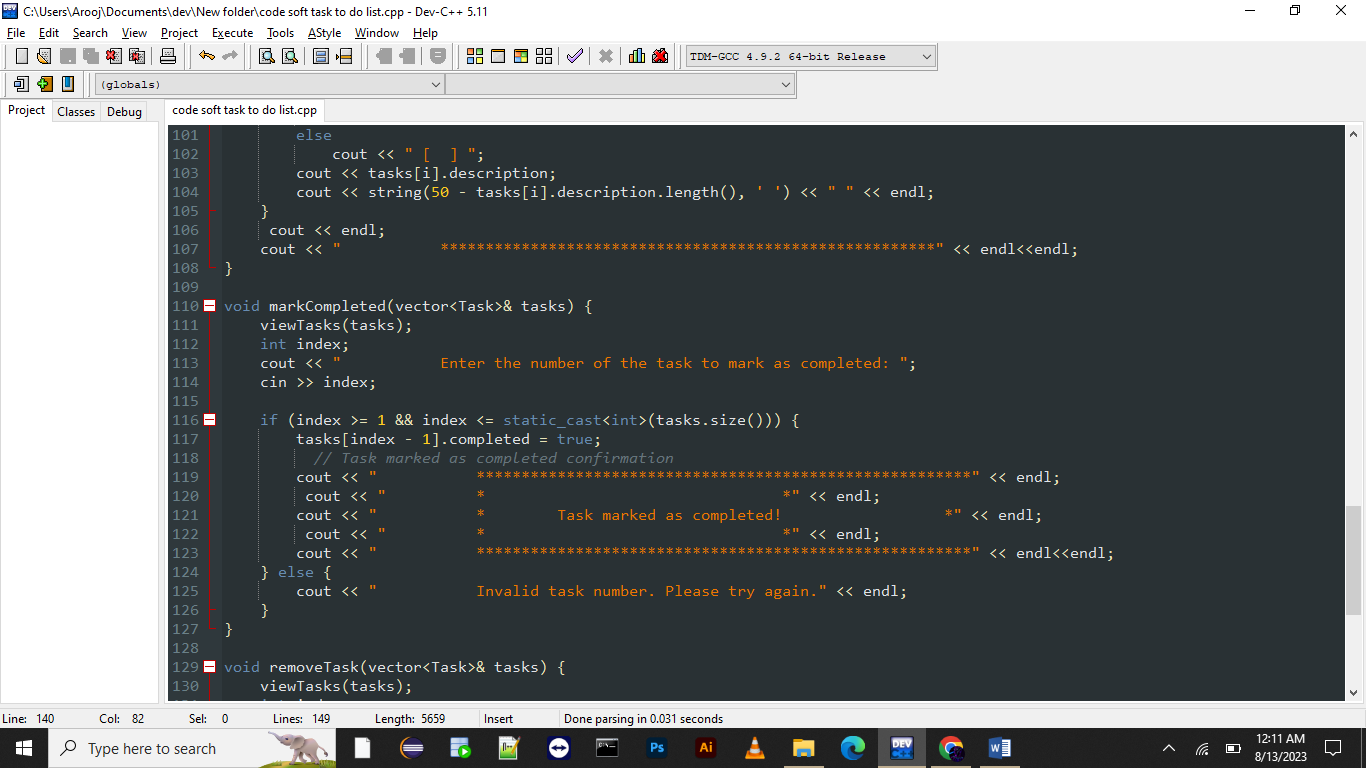
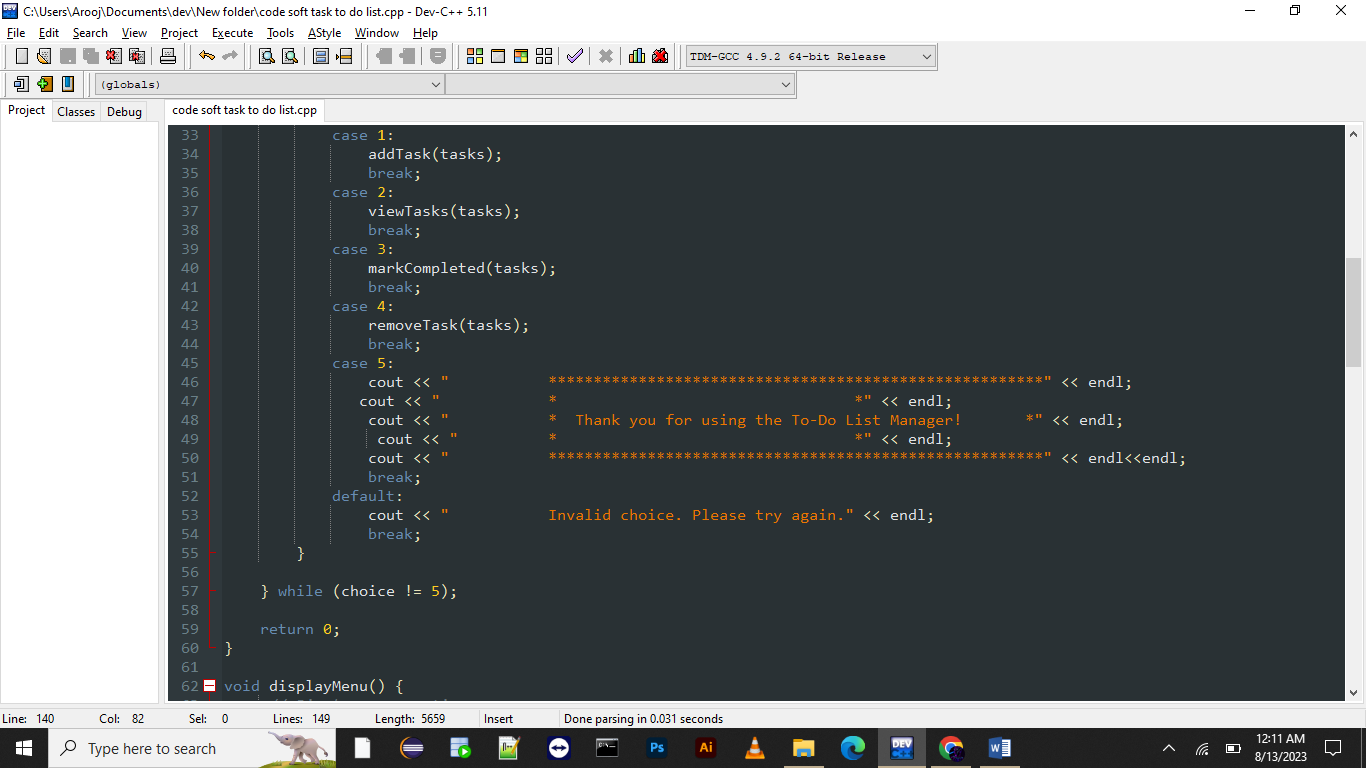
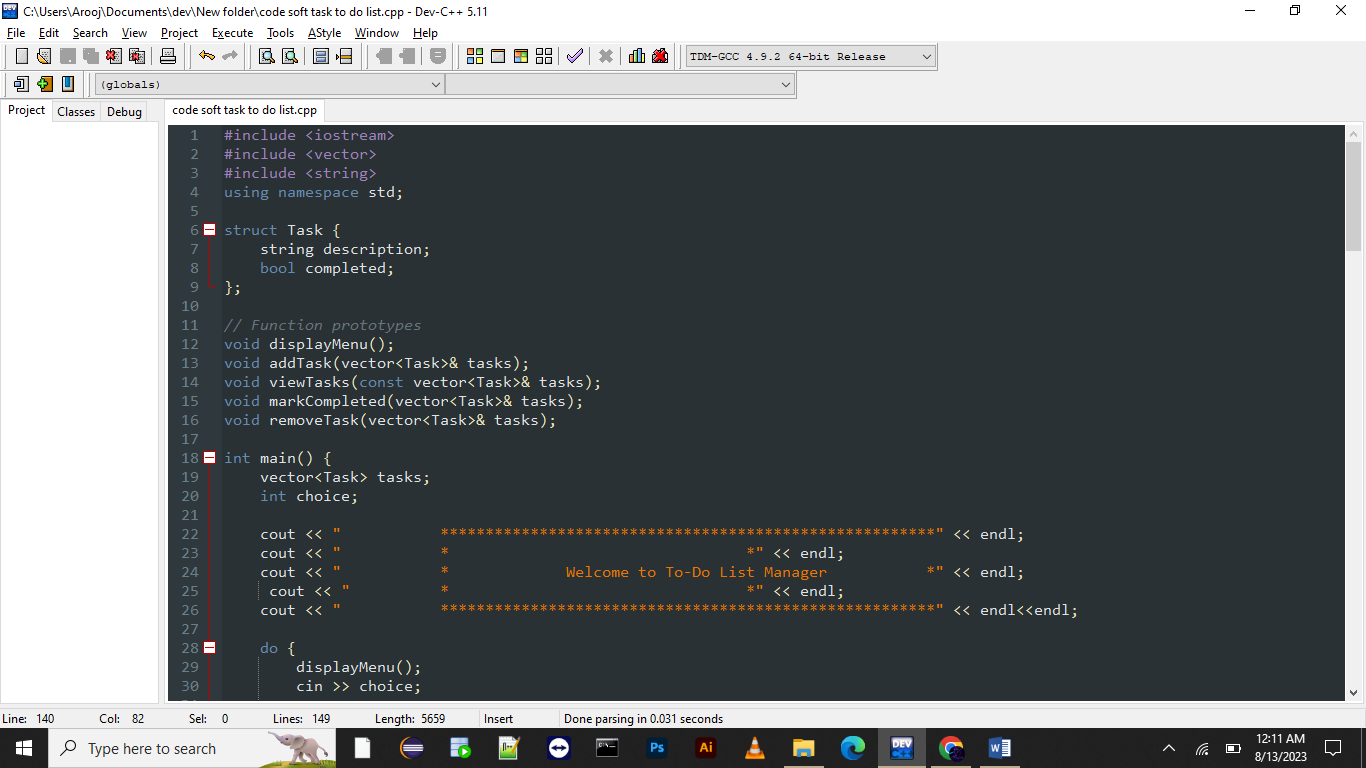
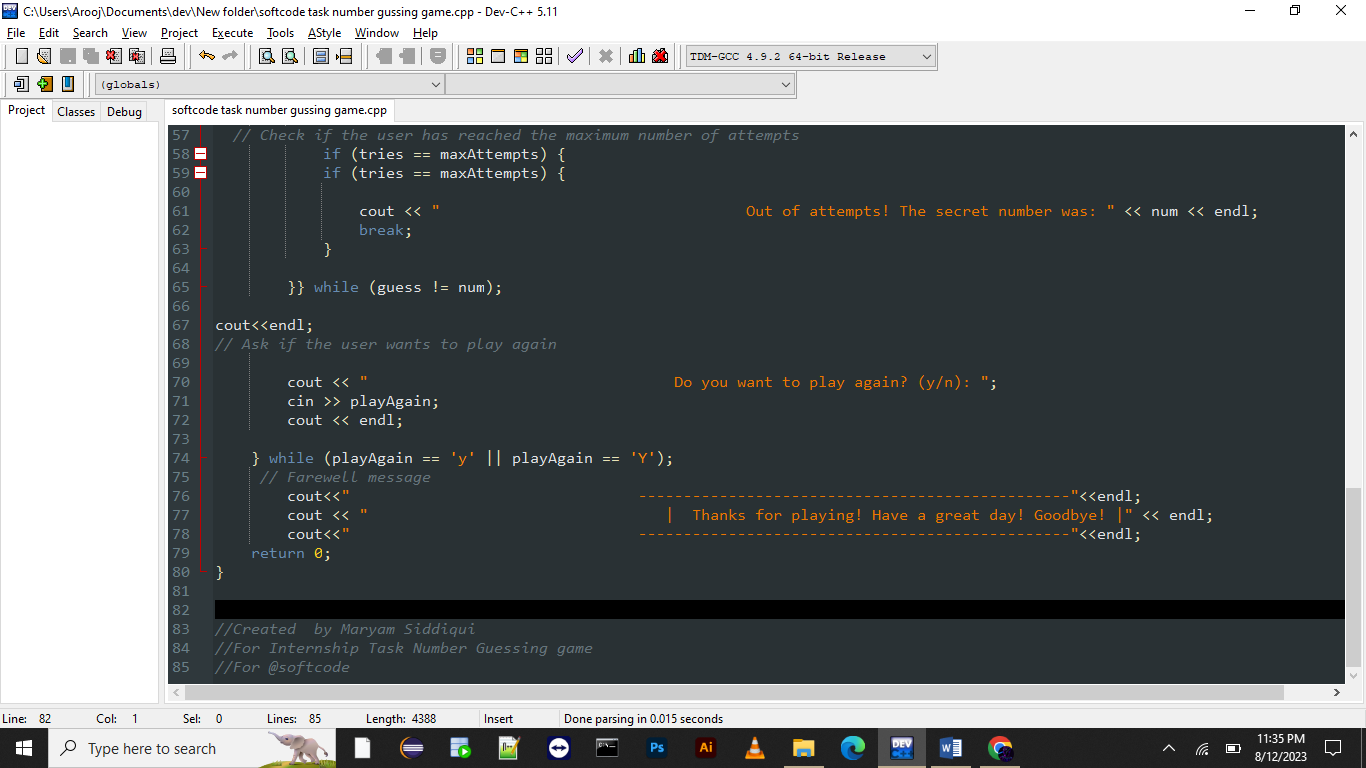
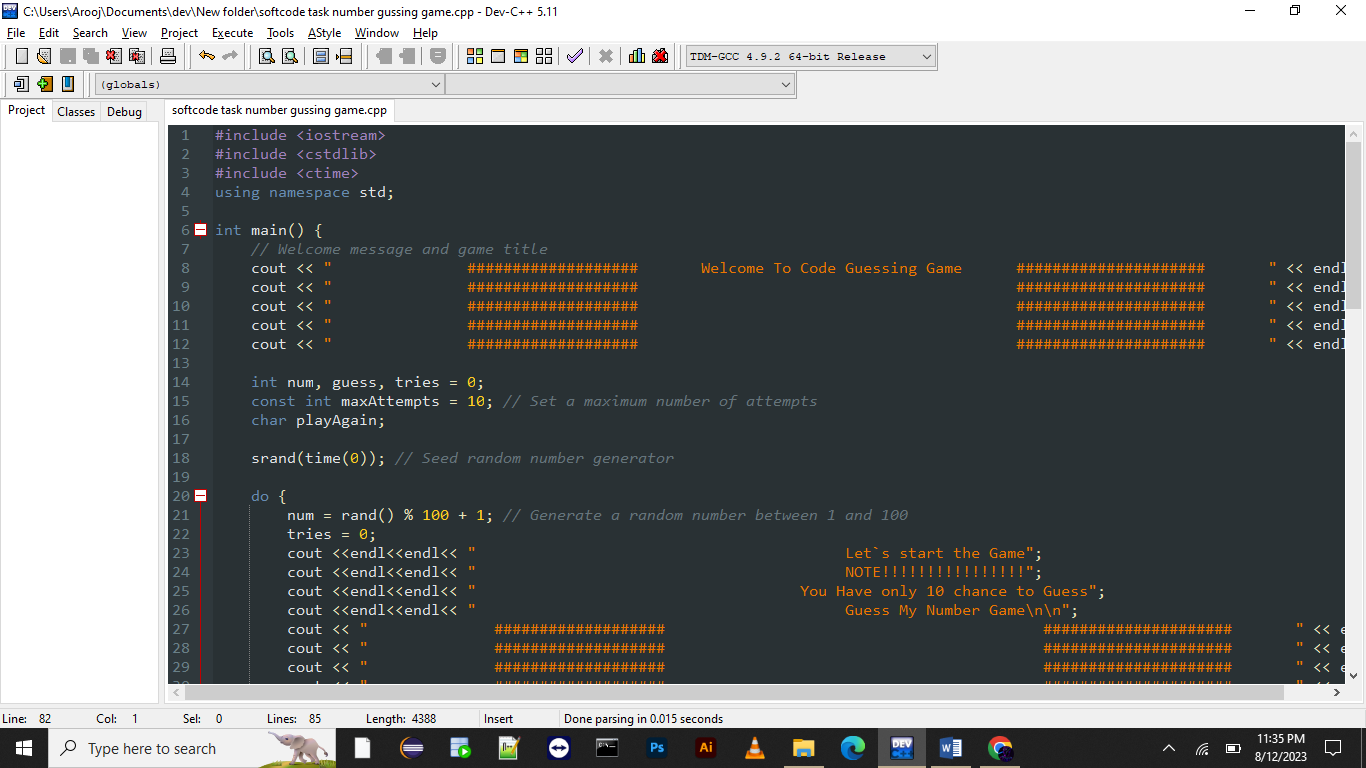
}

}

**//Created by Maryam Siddiqui**

**//For Internship Task To-do manager**

**//For @softcode**

****