TO DO LIST MANAGER

Project submitted to the

SRM University – AP, Andhra Pradesh

For the partial fulfilment of the requirements to award the degree of

Bachelor of Technology/Master of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

G.ROHITH SAI---AP22110010954

K.BHARGAV---AP22110010971

ERSHAD ALI---AP22110010941

V.YETHIN---AP22110010935

Under the Guidance of

KAVITHA MAM

SRM University–AP
Neerukonda, Mangalagiri, Guntur
Andhra Pradesh – 522 240

[NOVEMBER, 2023]

Certificate

Date: 21/11/2023

This is to certify that the work present in this Project entitled “TO DO LIST MANAGER” has been carried out by the group, [G. ROHITH, K. BHARGAV, E.ALI, V. YETHIN] under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in School of Engineering and Sciences.

Supervisor

(Signature)

Dr. Kavitha

CSE Department,

SRM UNIVERSITY, AP.

Table of Contents

Abstract

Our project is a “TO DO LIST MANAGER”. In this busy atmospheric world, people usually tend to forget about their important tasks which they need to perform and some of the daily needs. There’s nothing to blame on us it’s the atmosphere we live in.

So, in order to find a solution to this problem we need something which reminds our important tasks and things which need to be done on a daily basis without fail, so that we don’t forget anything and be productive in life and move further

“To do list manager” does the work of reminding us our tasks and daily routine on a daily basis and makes sure that we are not missing out on anything.

Introduction

In this project “TO DO LIST MANAGER” the user will be provided with an option to add a task, which he/she wants to get notified. When the user enters the taskthe program takes the input and stores it. There is no limit for the tasks.

User can also write a short note about the task which he wants to get remembered while doing the task. user can also set a remainder to the task about at what time it should notifty.

After completing the task when the user marks it as completed it will be deleted from the list. In this way the TO DO LIST MANAGER works Helping user to remember all their important tasks and their daily needs.

Methodology

Design:

In this project there will be 2 classes, one for storing the content of a task such as description and due date and another class for performing insertion and deletion operations using member functions.

Class Task ():

This class contains of 2 data members which are description of the task which is a string and the due date of the task which is an integer. Due date is stored in format of YYYY MM DD.

Class ToDoList ():

This class contains of a private attribute which is a vector of type < ToDoList> to store all the tasks and 3 member functions to add a task , display all the tasks ,marking a task as completed and removing a task.

- In the main function a do while is written which displays all choices and asks user to enter and choice and directs to the function and performs the task and when 0 is entered the program stops executing

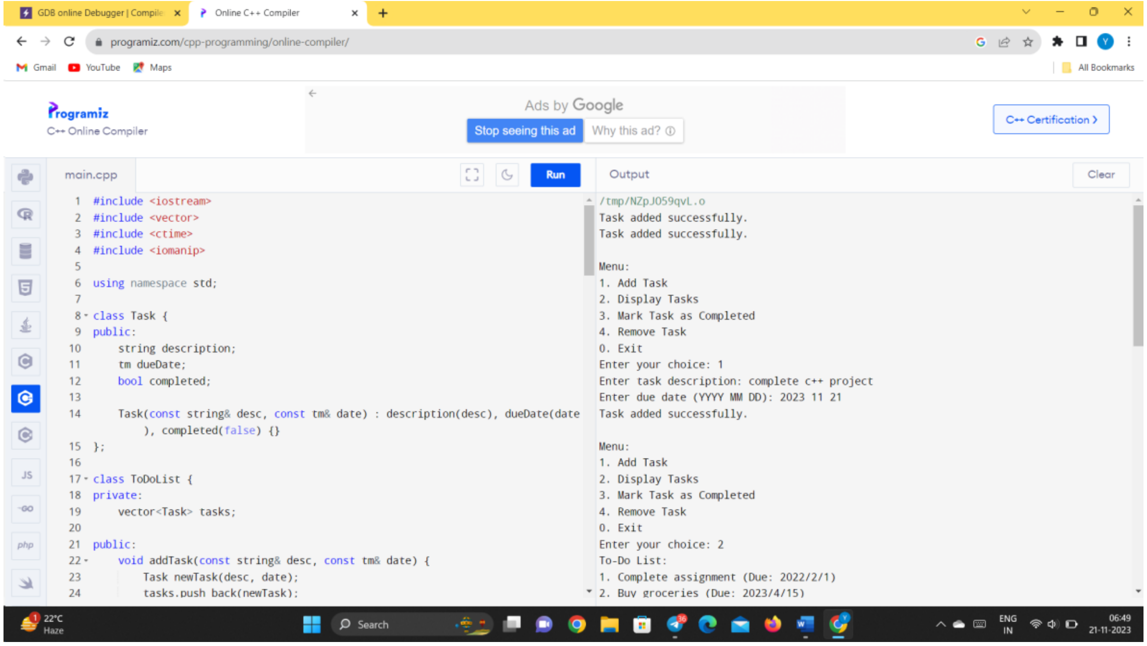
Implementation:

CODE:

```
#include <iostream>
#include <vector>
#include <ctime>
#include <iomanip>

using namespace std;

class Task {
public:
    string description;
    tm dueDate;
    bool completed;

    Task(const string& desc, const tm& date) : description(desc), dueDate(date),
completed(false) {}
};

class ToDoList {
private:
    vector<Task> tasks;

public:
    void addTask(const string& desc, const tm& date) {
        Task newTask(desc, date);
        tasks.push_back(newTask);
        cout << "Task added successfully.\n";
    }

    void displayTasks() const {
        if (tasks.empty()) {
            cout << "No tasks in the to-do list.\n";
        } else {
            cout << "To-Do List:\n";
            for (size_t i = 0; i < tasks.size(); ++i) {
                const auto& task = tasks[i];
                cout << i + 1 << ". " << task.description;
                cout << " (Due: " << task.dueDate.tm_year + 1900 << "/" <<
task.dueDate.tm_mon + 1 << "/"
                      << task.dueDate.tm_mday << ")";
                if (task.completed) {
                    cout << " [Completed]";
                }
                cout << "\n";
            }
        }
    }

    void markTaskAsCompleted(size_t index) {
        if (index < tasks.size()) {
            tasks[index].completed = true;
            cout << "Task marked as completed.\n";
        } else {
            cout << "Invalid task index.\n";
        }
    }

    void removeTask(size_t index) {
        if (index < tasks.size()) {
            tasks.erase(tasks.begin() + index);
            cout << "Task removed successfully.\n";
        } else {
            cout << "Invalid task index.\n";
        }
    }
};

int main() {
    ToDoList toDoList;

    // Example tasks
    tm dueDate1 = {0, 0, 0, 1, 1, 122};  // Due on January 1, 2023
    tm dueDate2 = {0, 0, 0, 15, 3, 123}; // Due on March 15, 2023

    toDoList.addTask("Complete assignment", dueDate1);
    toDoList.addTask("Buy groceries", dueDate2);

    int choice;
    do {
        cout << "\nMenu:\n";
        cout << "1. Add Task\n";
        cout << "2. Display Tasks\n";
        cout << "3. Mark Task as Completed\n";
        cout << "4. Remove Task\n";
        cout << "0. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1: {
                string desc;
                cout << "Enter task description: ";
                cin.ignore(); // Clear input buffer
                getline(cin, desc);

                cout << "Enter due date (YYYY MM DD): ";
                int year, month, day;
                cin >> year >> month >> day;

                tm dueDate = {};
                dueDate.tm_year = year - 1900; // Adjust for the year (tm_year is years since
1900)
                dueDate.tm_mon = month - 1;    // Adjust for the month (tm_mon is 0-based)
                dueDate.tm_mday = day;

                toDoList.addTask(desc, dueDate);
                break;
            }
            case 2:
                toDoList.displayTasks();
                break;
            case 3: {
                size_t index;
                cout << "Enter the index of the task to mark as completed: ";
                cin >> index;
                toDoList.markTaskAsCompleted(index - 1);
                break;
            }
            case 4: {
                size_t index;
                cout << "Enter the index of the task to remove: ";
                cin >> index;
                toDoList.removeTask(index - 1);
                break;
            }
            case 0:
                cout << "Exiting the program.\n";
                break;
            default:
                cout << "Invalid choice. Try again.\n";
        }
    } while (choice != 0);

    return 0;
}
```

Discussion

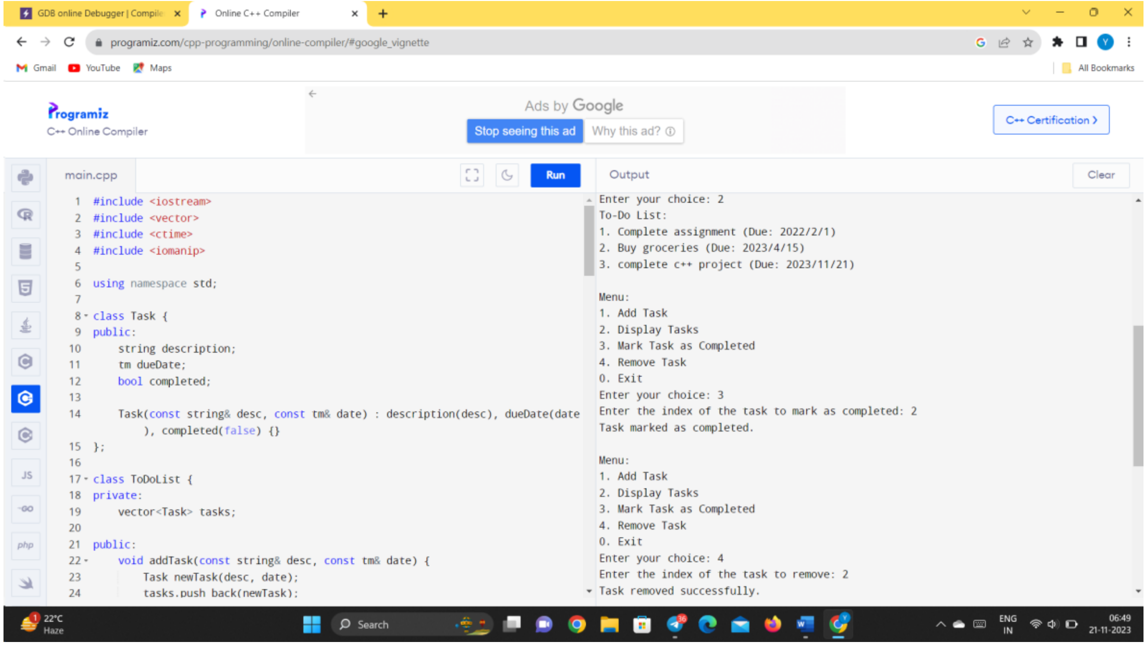
Output:



```cpp
#include <iostream>
#include <vector>
#include <ctime>
#include <iomanip>

using namespace std;

class Task {
public:
    string description;
    tm dueDate;
    bool completed;

    Task(const string& desc, const tm& date) : description(desc), dueDate(date
        ), completed(false) {}
};

class ToDoList {
private:
    vector<Task> tasks;

public:
    void addTask(const string& desc, const tm& date) {
        Task newTask(desc, date);
        tasks.push_back(newTask);
```

```
/tmp/NZpJO59qvL.o
Task added successfully.
Task added successfully.

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 1
Enter task description: complete c++ project
Enter due date (YYYY MM DD): 2023 11 21
Task added successfully.

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 2
To-Do List:
1. Complete assignment (Due: 2022/2/1)
2. Buy groceries (Due: 2023/4/15)
```

*4 choices are given,When the user selects to add a task the program takes the input of the task description and due date and adds it

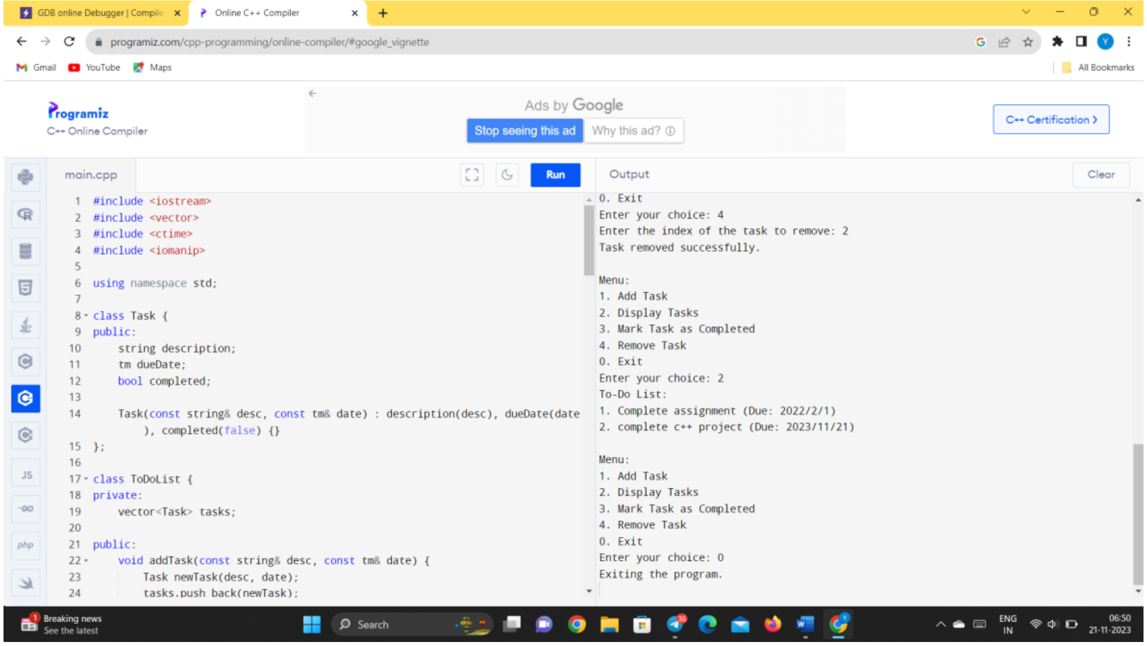


GDB online Debugger | Compile
Online C++ Compiler
programiz.com/cpp-programming/online-compiler/#google_vignette
Programiz
C++ Online Compiler
C++ Certification
main.cpp
Run
Output
Clear

```cpp
#include <iostream>
#include <vector>
#include <ctime>
#include <iomanip>

using namespace std;

class Task {
public:
    string description;
    tm dueDate;
    bool completed;

    Task(const string& desc, const tm& date) : description(desc), dueDate(date
        ), completed(false) {}
};

class ToDoList {
private:
    vector<Task> tasks;

public:
    void addTask(const string& desc, const tm& date) {
        Task newTask(desc, date);
        tasks.push_back(newTask);
```

```
Enter your choice: 2
To-Do List:
1. Complete assignment (Due: 2022/2/1)
2. Buy groceries (Due: 2023/4/15)
3. complete c++ project (Due: 2023/11/21)

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 3
Enter the index of the task to mark as completed: 2
Task marked as completed.

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 4
Enter the index of the task to remove: 2
Task removed successfully.
```

*When option 2 is chosen all the tasks are displayed successfilly .



```cpp
#include <iostream>
#include <vector>
#include <ctime>
#include <iomanip>

using namespace std;

class Task {
public:
    string description;
    tm dueDate;
    bool completed;

    Task(const string& desc, const tm& date) : description(desc), dueDate(date
        ), completed(false) {}
};

class ToDoList {
private:
    vector<Task> tasks;

public:
    void addTask(const string& desc, const tm& date) {
        Task newTask(desc, date);
        tasks.push_back(newTask);
```

```
0. Exit
Enter your choice: 4
Enter the index of the task to remove: 2
Task removed successfully.

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 2
To-Do List:
1. Complete assignment (Due: 2022/2/1)
2. complete c++ project (Due: 2023/11/21)

Menu:
1. Add Task
2. Display Tasks
3. Mark Task as Completed
4. Remove Task
0. Exit
Enter your choice: 0
Exiting the program.
```

*When option 3 is chosen it asks the index number of the task so that it marks that particular task as completed

*When option 4 is chosen, programs asks user to enter index of task so that it deletes the task

*When 0 is entered the program will be stopped coming out of the while loop.

Conclusion:

 The conclusion of the proposed system is based on users need. This system is developed in considering all the issues faced by the users.

 This implementation of “TO DO LIST MANAGER” is done to help and solve problems of people regarding forgetting their tasks.

 This improves the productivity of the user helping him not to overdue any of the tasks.

 This will help the user to improve the overall growth aspect including carrier and health.

*****THE END *****