University of Liberal Arts Bangladesh

(Open Ended Lab Report)

Course Code	**:**	**CSE2202**
Course Title	**:**	**Algorithm Lab**
Section	**:**	**05**
Submitted to	**:**	**Fatema khan**
Submitted by	**:**	**Kawser Ahmmed**
ID	**:**	**231014036**

Problem: Suppose, you've been hired by a Project Management Consultancy firm that handles multiple
projects simultaneously. Each project requires various resources such as manpower, equipment,
and funds. The challenge is to allocate resources optimally across different projects to maximize
overall profitability while meeting project deadlines and resource constraints. Your task is to develop a resource allocation system to address the firm's challenges. The system
should consider various factors such as project profitability, resource requirements, project
timelines, and available resources to recommend the best combination of projects to undertake
within the firm's capacity.

Identification

When it comes to choosing the best solution to fix constraints, dynamic programming is best for that kind of problem. In this problem, I have to select projects and allocate resources so that I can get the maximum profit so that's why I think dynamic program will be the best solution and therse Multiple projects can be there at the same time, with the same duration and time, for this kind of collision, and select the maximum projects with a higher priority. I can also use activity selection to select the projects with the highest profits.

(dynamic programming , greedy method)

Discussion

First of all, I am using Compnies main funds to invest in multiple projects and two constrains costs per manpower cost and per equipment cost, so that after talking about project resources, I can calculate the profit and cost of every project.

For every project, I am taking the names manpower and eq and how much money I will get from that project, and then I calculate the profit by per manpower cost and per equipment cost. I am also using insertion sort to sort all projects by their ending time to find out if there are any collisions available. If I find any collisions by using custiom activity selection, I remove the collid projects with a lower profit (profit / duration) so that I can get the best one from the collid projects. After finding the maximum projects, the main work is to select the projects in such a way that I can get maximum profits with my company's fixed fund.

NOTE: I am taking projects for 1 year only after

Explanation

Here I am explaining my code objective and their working process:

- **Project Class:**

Defines a class named Project to represent each project. Each project has attributes such as name, duration, start time, end time, manpower, funds, profit, cost, and equipment. The constructor initializes these attributes based on user input.

- **Main Function:**

It starts by taking input for total funds, cost per manpower, and cost per equipment.
It provides a menu-driven interface for managing projects.
The menu options include:
Adding a project
Sorting projects based on end time
Removing colliding projects
Finding the maximum profit achievable with the given funds
Exiting the program
The user can choose options from the menu, and the program performs the corresponding actions.

- **Helper Functions:**

displayProjects: Displays the details of all projects in a tabular format.

insertionSort: Sorts the projects based on their end times using the insertion sort algorithm.
removeCollisions: Removes colliding projects from the list. If two projects overlap in time, it removes the project with lower profit per day.
maximumProfit: Finds the maximum profit achievable with the given funds by selecting the optimal combination of projects. It uses dynamic programming to solve the knapsack problem.
User Interaction:

The program interacts with the user through the console, prompting for input and displaying results accordingly.
It allows the user to add projects, manipulate project data, and find the optimal project combination for maximum profit.

Objective:
The main objective of this program is to assist in managing project resources efficiently by providing functionalities such as adding projects, sorting them based on end times, handling project collisions, and determining the optimal project combination for maximizing profit within the given constraints of funds and costs.

Code

```
#include <bits/stdc++.h>
using namespace std;

double cost_per_man;
double cost_per_equipment;

class Project
{
private:
   string name;
   float duration;
   float start_time;
   float end_time;
   float manpower;
   float funds;
   float profit;
   float cost;
   int equipment;

public:
   Project(string name, float duration, float start_time, float end_time, float
manpower, float funds, int equipment)
   {
      this->name = name;
      this->duration = duration;
      this->start_time = start_time;
      this->end_time = end_time;
      this->manpower = manpower;
      this->funds = funds;
      this->profit = funds - (manpower * cost_per_man + equipment *
cost_per_equipment);
      this->equipment = equipment;
      this->cost = (manpower * cost_per_man + equipment *
cost_per_equipment);
   }
```

```
    string getName() const { return name; }
    float getDuration() const { return duration; }
    float getStartTime() const { return start_time; }
    float getEndTime() const { return end_time; }
    float getManpower() const { return manpower; }
    float getFunds() const { return funds; }
    float getProfit() const { return profit; }
    float getCost() const { return cost; }
    int getEquipment() const { return equipment; }
};

void displayProjects(const vector<Project> &projects)
{
    cout << "-------------------------------------------------------------------------------------
-------------------------------------------------------------------\n";
    cout << "|       Project Name       | Duration(days)  | Start Time(month) | End
Time(month)  | Manpower(man)  | revinue+costs($)| Profit($)| cost |Equipments
|\n";
    cout << "-------------------------------------------------------------------------------------
-------------------------------------------------------------------\n";

    if (projects.empty())
    {
        cout << "| NO projects added\n";
    }
    else
    {
        for (const Project &project : projects)
        {
            cout << "| " << left << setw(25) << project.getName()
                 << "| " << setw(16) << project.getDuration()
                 << "| " << setw(19) << project.getStartTime()
                 << "| " << setw(17) << project.getEndTime()
                 << "| " << setw(15) << project.getManpower()
                 << "| " << setw(13) << project.getFunds()
                 << "| " << setw(9) << project.getProfit()
                 << " |" << setw(7) << project.getCost()
                 << "| " << setw(11) << project.getEquipment() << "|\n";
        }
    }

    cout << "-------------------------------------------------------------------------------------
-------------------------------------------------------------------\n";
}

void insertionSort(vector<Project> &projects)
```

```
{
   for (size_t i = 1; i < projects.size(); ++i)
   {
      Project key = projects[i];
      int j = i - 1;

      while (j >= 0 && projects[j].getEndTime() > key.getEndTime())
      {
         projects[j + 1] = projects[j];
         --j;
      }
      projects[j + 1] = key;
   }
}

void removeCollisions(vector<Project> &projects)
{
   for (size_t i = 0; i < projects.size(); ++i)
   {
      for (size_t j = i + 1; j < projects.size(); ++j)
      {
         if (projects[i].getEndTime() > projects[j].getStartTime())
         {
            if ((projects[i].getProfit() / projects[i].getDuration()) <
(projects[j].getProfit() / projects[j].getDuration()))
            {
               projects.erase(projects.begin() + i);
            }
            else
            {
               projects.erase(projects.begin() + j);
            }
         }
      }
   }
}

float maximumProfit(const vector<Project> &projects, float totalFunds,
vector<bool> &selectedProjects)
{
   int n = projects.size();
   vector<vector<float>> dp(n + 1, vector<float>(totalFunds + 1, 0));

   for (int i = 1; i <= n; ++i)
   {
      for (int j = 0; j <= totalFunds; ++j)
```

```
        {
            if (projects[i - 1].getCost() <= j)
            {
                dp[i][j] = max(dp[i - 1][j], projects[i - 1].getProfit() + dp[i - 1][j -
projects[i - 1].getCost()]);
            }
            else
            {
                dp[i][j] = dp[i - 1][j];
            }
        }
    }

    int maxProfit = dp[n][totalFunds];
    int remainingFunds = totalFunds;
    for (int i = n; i > 0 && maxProfit > 0; --i)
    {
        if (maxProfit != dp[i - 1][remainingFunds])
        {
            selectedProjects[i - 1] = true;
            maxProfit -= projects[i - 1].getProfit();
            remainingFunds -= projects[i - 1].getCost();
        }
    }

    return dp[n][totalFunds];
}

int main()
{
    cout << "-------------------------------------------\n";
    cout << "OPTIMAL RESOURSE ALLOCATION MANAGMENTS
SYSTEM\n";
    cout << "_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-\n";
    vector<Project> projects;
    float totalFunds;

    cout << "\nCompanies contrains:\n";
    cout << "**********************\n";
    cout << "Enter company's total funds: ";
    cin >> totalFunds;

    cout << "Enter per manpower cost: ";
    cin >> cost_per_man;

    cout << "Enter per equipment cost: ";
```

```
    cin >> cost_per_equipment;

    int choice;
    do
    {
        cout << "-------------------------------------\n";
        cout << " projects allcocation managment       \n";
        cout << "-------------------------------------\n";
        cout << "\n1. Add project\n";
        cout << "2. Sort projects based on end time\n";
        cout << "3. Remove colliding projects\n";
        cout << "4. Find maximum profit\n";
        cout << "5. Exit\n";
        cout << "Enter your choice:\n>> ";
        cin >> choice;
        cout << "*****************************************\n";

        switch (choice)
        {
        case 1:
        {
            string name;
            float duration, start_time, end_time, manpower, funds;
            int equipment;

            cout << "Enter project name: ";
            cin >> name;
            cout << "Enter project duration (days): ";
            cin >> duration;
            cout << "Enter project start time (month): ";
            cin >> start_time;
            cout << "Enter project end time (month): ";
            cin >> end_time;
            cout << "Enter manpower: ";
            cin >> manpower;
            cout << "Enter project revenue with costs: ";
            cin >> funds;
            cout << "Enter equipments: ";
            cin >> equipment;

            projects.push_back(Project(name, duration, start_time, end_time,
manpower, funds, equipment));
            break;
        }
        case 2:
            insertionSort(projects);
```

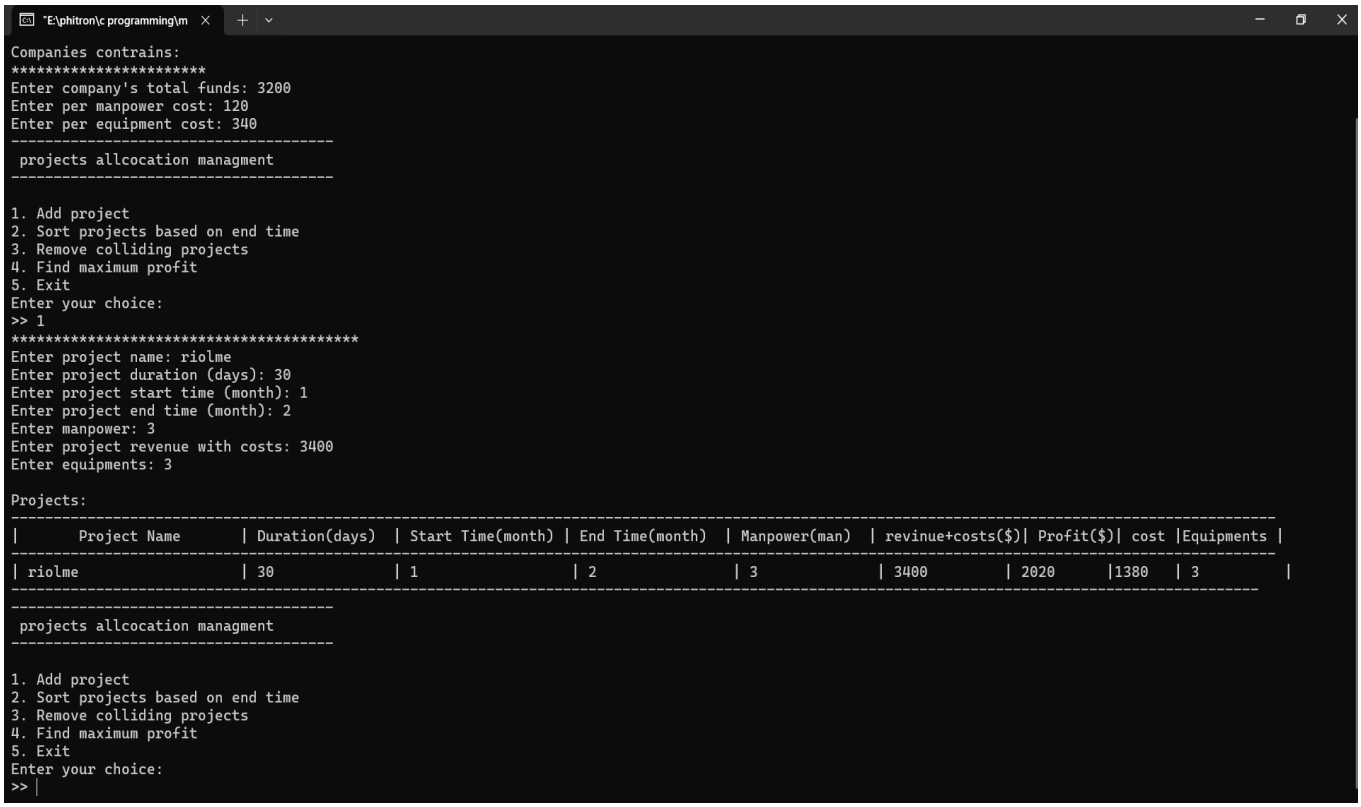
```
            cout << "Projects sorted based on end time.\n";
            break;
        case 3:
            removeCollisions(projects);
            cout << "Colliding projects removed.\n";
            break;
        case 4:
        {
            vector<bool> selectedProjects(projects.size(), false);
            float maxProfit = maximumProfit(projects, totalFunds,
selectedProjects);
            cout << "Maximum profit achievable: $" << maxProfit << endl;
            cout << "Selected projects to achieve maximum profit:\n";
            for (size_t i = 0; i < selectedProjects.size(); ++i)
            {
                if (selectedProjects[i])
                {
                    cout << projects[i].getName() << endl;
                }
            }
            break;
        }
        case 5:
            cout << "Exiting.\n";
            break;
        default:
            cout << "Invalid choice! Please enter a valid option.\n";
        }

        if (choice != 5)
        {
            cout << "\nProjects:\n";
            displayProjects(projects);
        }
    } while (choice != 5);

    return 0;
}
```

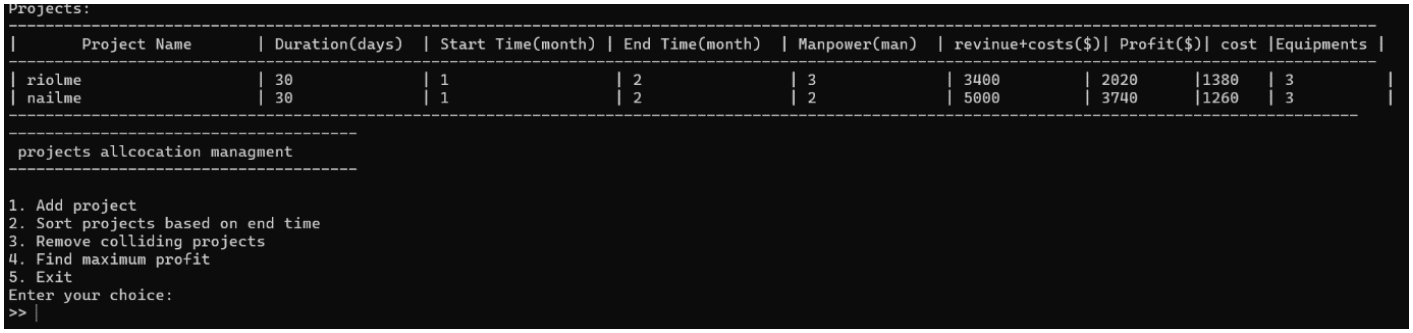
Output Analysis

1. Adding projects on table:

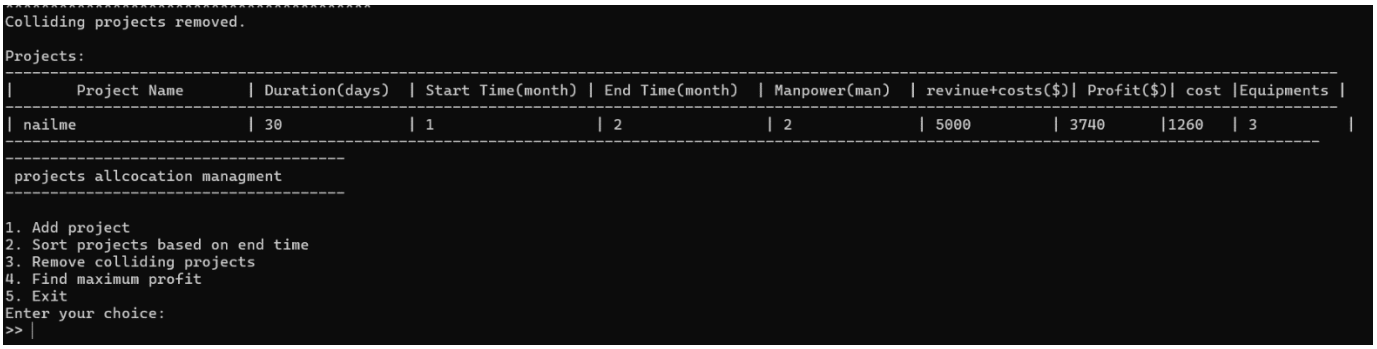


2. Removeing collisions of projects:

Before removing collisions:



After removing collisoions and chosing the best one from collisons:



3. Sorting projects based on their ending time:
   Unsorted table:

```
Projects:
-------------------------------------------------------------------------------------------------------------------------------------------------
|       Project Name       | Duration(days)  | Start Time(month) | End Time(month)  | Manpower(man)  | revinue+costs($)| Profit($)| cost |Equipments |
-------------------------------------------------------------------------------------------------------------------------------------------------
| naile                    | 30              | 1                 | 2                | 3              | 3400          | 2020      |1380   | 3          |
| gorgopull                | 60              | 6                 | 8                | 2              | 3000          | 1400      |1600   | 4          |
| pocinki                  | 20              | 3                 | 4                | 4              | 2000          | -180      |2180   | 5          |
| trastiolas               | 56              | 9                 | 11               | 2              | 7000          | 5060      |1940   | 5          |
-------------------------------------------------------------------------------------------------------------------------------------------------

---------------------------------------
 projects allcocation managment
---------------------------------------

1. Add project
2. Sort projects based on end time
3. Remove colliding projects
4. Find maximum profit
5. Exit
Enter your choice:
>>
```

Sorted table :

```
Projects:
-------------------------------------------------------------------------------------------------------------------------------------------------
|       Project Name       | Duration(days)  | Start Time(month) | End Time(month)  | Manpower(man)  | revinue+costs($)| Profit($)| cost |Equipments |
-------------------------------------------------------------------------------------------------------------------------------------------------
| naile                    | 30              | 1                 | 2                | 3              | 3400          | 2020      |1380   | 3          |
| pocinki                  | 20              | 3                 | 4                | 4              | 2000          | -180      |2180   | 5          |
| gorgopull                | 60              | 6                 | 8                | 2              | 3000          | 1400      |1600   | 4          |
| trastiolas               | 56              | 9                 | 11               | 2              | 7000          | 5060      |1940   | 5          |
-------------------------------------------------------------------------------------------------------------------------------------------------

---------------------------------------
 projects allcocation managment
---------------------------------------

1. Add project
2. Sort projects based on end time
3. Remove colliding projects
4. Find maximum profit
5. Exit
Enter your choice:
>>
```

4.finding the maximum profit (according to the cost selecting the best profitable projects and selected projects name :

```
Maximum profit achievable: $5060
Selected projects to achieve maximum profit:
trastiolas

Projects:
-------------------------------------------------------------------------------------------------------------------------------------------------
|       Project Name       | Duration(days)  | Start Time(month) | End Time(month)  | Manpower(man)  | revinue+costs($)| Profit($)| cost |Equipments |
-------------------------------------------------------------------------------------------------------------------------------------------------
| naile                    | 30              | 1                 | 2                | 3              | 3400          | 2020      |1380   | 3          |
| pocinki                  | 20              | 3                 | 4                | 4              | 2000          | -180      |2180   | 5          |
| gorgopull                | 60              | 6                 | 8                | 2              | 3000          | 1400      |1600   | 4          |
| trastiolas               | 56              | 9                 | 11               | 2              | 7000          | 5060      |1940   | 5          |
-------------------------------------------------------------------------------------------------------------------------------------------------

---------------------------------------
 projects allcocation managment
---------------------------------------

1. Add project
2. Sort projects based on end time
3. Remove colliding projects
4. Find maximum profit
5. Exit
Enter your choice:
>>
```

Here we can see this program selected only one item. The company have fund 3400$ in that amount to get maximum profit have to select item “trastiolas” wich profit is best for companies (5060$)

Calculating cost = (per manpower cost*maxpower+per eqapment*equipment);

Profit= revinue - (per manpower cost*maxpower+per eqapment*equipment);