Name:-Dikesh Ganboi
Roll NO:- A36

```
#include <iostream>
#include <algorithm>
#include <vector>

using namespace std;

// Structure to represent a job
struct Job {
    int id;       // Job ID
    int profit;   // Profit associated with the job
    int deadline; // Deadline by which the job needs to be completed
};

// Function to compare jobs based on profit (in descending order)
bool compare(Job& a, Job& b) {
    return a.profit > b.profit;
}

// Function to find the maximum deadline
int findMaxDeadline(vector<Job>& jobs) {
    int maxDeadline = 0;
    for (auto& job : jobs) {
        maxDeadline = max(maxDeadline, job.deadline);
    }
    return maxDeadline;
}

// Function to perform job sequencing using greedy approach
pair<int, vector<int>> jobSequencing(vector<Job>& jobs) {
    // Sort jobs based on profit in descending order
    sort(jobs.begin(), jobs.end(), compare);

    // Find maximum deadline
    int maxDeadline = findMaxDeadline(jobs);

    // Initialize schedule array with -1 (indicating slot is empty)
    vector<int> schedule(maxDeadline, -1);

    // Iterate through each job and assign it to a time slot
    int totalProfit = 0;
    vector<int> selectedJobs;
```

```cpp
    for (auto& job : jobs) {
        int deadline = job.deadline;
        // Find available slot before the deadline
        while (deadline > 0 && schedule[deadline - 1] != -1) {
            deadline--;
        }
        // If slot available, assign job to slot
        if (deadline > 0) {
            schedule[deadline - 1] = job.id;
            totalProfit += job.profit;
            selectedJobs.push_back(job.id);
        }
    }

    return {totalProfit, selectedJobs};
}

int main() {
    // Example usage
    vector<Job> jobs = {{1, 20, 4}, {2, 15, 2}, {3, 10, 3}, {4, 5, 3}};

    // Find maximum profit and list of selected jobs
    auto result = jobSequencing(jobs);
    int maxProfit = result.first;
    vector<int> selectedJobs = result.second;

    // Output results
    cout << "Maximum Profit: " << maxProfit << endl;
    cout << "Selected Jobs: ";
    for (int job : selectedJobs) {
        cout << job << " ";
    }
    cout << endl;

    return 0;
}
```

OUTPUT:-

```
PS C:\Users\HP\Desktop\DAA EXperiment> cd "c:\Users\HP\Desktop\DAA
EXperiment\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o
tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }

Maximum Profit: 50
Selected Jobs: 1 2 3 4
```