\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Assignment No: 2**

**Title Name: Write a program to implement Job sequencing with deadlines using a greedy method.**

**Name**: **Prasanna Satish Kulkarni**

**Class**: **BE** **Div: BE-I** **Batch**: **B**

**Exam Seat No/Roll No: 405A048**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Program:**

#include <algorithm>

#include <iostream>

using namespace std;

struct Job {

char id; // Job Id

int dead; // Deadline of job

int profit; // Profit if job is over before or on deadline

};

bool comparison(Job a, Job b)

{

return (a.profit > b.profit);

}

// Returns minimum number of platforms required

void printJobScheduling(Job arr[], int n)

{

// Sort all jobs according to decreasing order of profit

sort(arr, arr + n, comparison);

int result[n]; // To store result (Sequence of jobs)

bool slot[n]; // To keep track of free time slots

// Initialize all slots to be free

for (int i = 0; i < n; i++)

slot[i] = false;

// Iterate through all given jobs

for (int i = 0; i < n; i++) {

// Find a free slot for this job (Note that we start

// from the last possible slot)

for (int j = min(n, arr[i].dead) - 1; j >= 0; j--) {

// Free slot found

if (slot[j] == false) {

result[j] = i; // Add this job to result

slot[j] = true; // Make this slot occupied

break;

}

}

}

// Print the result

int sum=0;

for (int i = 0; i < n; i++)

if (slot[i]){

cout << arr[result[i]].id << " ";

sum=sum+ arr[result[i]].profit;

} cout << "\n Maximum Profit: " << sum << " ";

}

int main()

{

Job arr[] = { { 'a', 2, 100 },

{ 'b', 1, 19 },

{ 'c', 2, 27 },

{ 'd', 1, 25 },

{ 'e', 3, 15 } };

int n = sizeof(arr) / sizeof(arr[0]);

cout << "Following is maximum profit sequence of jobs ";

cout << "\n Sequence of jobs: ";

// Function call

printJobScheduling(arr, n);

return 0;

}

**Output:**

Following is maximum profit sequence of jobs

Sequence of jobs: c a e

Maximum Profit: 142