**Job Scheduling  
Source Code:**

#include <stdio.h>

#define MAX 100

typedef struct Job {

char id[5];

int deadline;

int profit;

} Job;

void jobSequencingWithDeadline(Job jobs[], int n);

int minValue(int x, int y) {

if(x < y) return x;

return y;

}

int main(void) {

//variables

int i, j;

//jobs with deadline and profit

Job jobs[5] = {

{"j1", 2, 60},

{"j2", 1, 100},

{"j3", 3, 20},

{"j4", 2, 40},

{"j5", 1, 20},

};

//temp

Job temp;

//number of jobs

int n = 5;

//sort the jobs profit wise in descending order

for(i = 1; i < n; i++) {

for(j = 0; j < n - i; j++) {

if(jobs[j+1].profit > jobs[j].profit) {

temp = jobs[j+1];

jobs[j+1] = jobs[j];

jobs[j] = temp;

}

}

}

printf("%10s %10s %10s\n", "Job", "Deadline", "Profit");

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

printf("%10s %10i %10i\n", jobs[i].id, jobs[i].deadline, jobs[i].profit);

}

jobSequencingWithDeadline(jobs, n);

return 0;

}

void jobSequencingWithDeadline(Job jobs[], int n) {

//variables

int i, j, k, maxprofit;

//free time slots

int timeslot[MAX];

//filled time slots

int filledTimeSlot = 0;

//find max deadline value

int dmax = 0;

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

if(jobs[i].deadline > dmax) {

dmax = jobs[i].deadline;

}

}

//free time slots initially set to -1 [-1 denotes EMPTY]

for(i = 1; i <= dmax; i++) {

timeslot[i] = -1;

}

printf("dmax: %d\n", dmax);

for(i = 1; i <= n; i++) {

k = minValue(dmax, jobs[i - 1].deadline);

while(k >= 1) {

if(timeslot[k] == -1) {

timeslot[k] = i-1;

filledTimeSlot++;

break;

}

k--;

}

//if all time slots are filled then stop

if(filledTimeSlot == dmax) {

break;

}

}

//required jobs

printf("\nRequired Jobs: ");

for(i = 1; i <= dmax; i++) {

printf("%s", jobs[timeslot[i]].id);

if(i < dmax) {

printf(" --> ");

}

}

//required profit

maxprofit = 0;

for(i = 1; i <= dmax; i++) {

maxprofit += jobs[timeslot[i]].profit;

}

printf("\nMax Profit: %d\n", maxprofit);

}

**OUTPUT-**

