Experiment No.9

Aim: Implement Greedy Algorithm for Job Sequencing with Deadlines.

Theoretical Background:

In job sequencing problem, the objective is to find a sequence of jobs, which is completed within their deadlines and gives maximum profit.

Solution Let us consider, a set of n given jobs which are associated with deadlines and profit is earned, if a job is completed by its deadline. These jobs need to be ordered in such a way that there is maximum profit.

It may happen that all of the given jobs may not be completed within their deadlines. Assume, deadline of i th job Ji is di and the profit received from this job is pi. Hence, the optimal solution of this algorithm is a feasible solution with maximum profit. Thus, D(i)>0 for 1⩽i⩽n Initially, these jobs are ordered according to profit, i.e. p1⩾p2⩾p3⩾...⩾pn

Program:

```
#include <stdio.h>

#include <stdlib.h>

#include<conio.h>

typedef struct {

   char id;

   int deadline;

   int profit;

}Job;

int compareJob(const void *a, const void *b){

   //Will return true if a's profit > b's profit

   //else will return false

   return ((Job*)a)->profit - ((Job*)a)->profit;

}



void bestJob(Job jobs[],int sizeOfJobs){
   char jobsToDo[5]= {'\0'}; //Assign every element of array to '\0'-Only works
in few compilers
   //If above line do not work use for loop to assign '\0' to every element
   int i, k;
   for(i=0; i< sizeOfJobs; i++){
            k= jobs[i].deadline-1;
         //Searching for empty date nearest to deadline backwards
         while(jobsToDo[k] != '\0' && k >= 0){
            k--;
         }
         if(k != -1)
            jobsToDo[k]= jobs[i].id;
   }
   printf("Best order and jobs to do is \n");
   k=0;
   while(jobsToDo[k] != '\0'){
         printf("%c ",jobsToDo[k]);
         k++;
   }
}

void display(Job jobs[],int n){
   int i;
   printf("Job Id: \t");
   for(i=0; i<n; i++){
         printf("%c \t",jobs[i].id);
   }
   printf("\n");
   printf("Job Deadline: \t");
   for(i=0; i<n; i++){
         printf("%d \t",jobs[i].deadline);
   }
   printf("\n");
   printf("Job Profit: \t");
   for(i=0; i<n; i++){
         printf("%d \t",jobs[i].profit);
   }
   printf("\n");
}
int main()
{    clrscr();
   Job jobs[]=  {{'w', 1, 19}, {'v', 2, 100}, {'x', 2, 27},
                 {'y', 1, 25}, {'z', 3, 15}};
   display(jobs,5);
   //sorting jobs[] w.r.t profit
   qsort(jobs,5,sizeof(jobs[0]),compareJob);
   bestJob(jobs,5);
   getch();
   return 0;
}
```

Output:

Conclusion:

Thus, in this experiment we have studied about implementing job sequencing with deadlines.