Practical no: 4

Name: Saloni Vishwakarma
Batch-Roll no: C1-13
Subject: DAA Lab

Aim: a) There is one meeting room in a small firm. There is N number of meetings that has to be conducted for a single day, meetings in the form of (S[i], F[i]) where S[i] is the start time of meeting i and F[i] is the finish time of meeting i. The task is to find the maximum number of meetings that can be accommodated in the meeting room. Print all meeting numbers in ascending order.
(Activity Selection Algorithm)

b) Suppose you are given 10 types of vegetables which weigh different and the total weight of 10 vegetables is around 25 kg. You have a bag that can hold a maximum of 10 kg of vegetables. Here 10 kg limit is also known as Knapsack bag or Knapsack weight. How will you choose the vegetables for your bag?
(Knapsack Algorithm)

1) Activity Selection Algorithm (Code and Output):

```
#include<stdio.h>
int main(){
   int n;
   printf("\n Enter the number of activities: ");
   scanf("%d",&n);
   int i,j,t;
   int index[n],start[n],fin[n];
   printf("\n Enter the process number: ");
   for(i=0;i<n;i++)
   scanf("%d",&index[i]);
   printf("\n Enter the start time: ");
   for(i=0;i<n;i++)
   scanf("%d",&start[i]);
   printf("\n Enter the finish time: ");
   for(i=0;i<n;i++)
   scanf("%d",&fin[i]);
   for(i=0;i<n;i++)
  {
```

```
    for(j=i+1;j<n;j++)
   {
       if(fin[j]<fin[i])
      {
          t=fin[j];
          fin[j]=fin[i];
          fin[i]=t;
          t=start[j];
          start[j]=start[i];
          start[i]=t;

          t=index[j];
          index[j]=index[i];
          index[i]=t;
       }
     }
}
printf("\n After sorting:");
printf("\n Start array is: ");
for(i=0;i<n;i++)
printf("%d ",start[i]);
printf("\n Finish array is: ");
for(i=0;i<n;i++)
printf("%d ",fin[i]);
printf("\n Index array is: ");
for(i=0;i<n;i++)
printf("%d ",index[i]);
printf ("\n\n Following activities are selected: ");
i = 0;
int c=1;
printf("%d ", index[i]);
for (j = 1; j < n; j++)
{
   if (start[j] >= fin[i])
  {
      printf ("%d ", index[j]);
        i = j;
        c++;
   }
}
```

```
  printf("\n %d activities are selected in total",c);
}
```

```
Enter the number of activities: 6

Enter the process number: 1 2 3 4 5 6

Enter the start time: 0 3 1 5 5 8

Enter the finish time: 6 4 2 9 7 9

After sorting:
Start array is: 1 3 0 5 5 8
Finish array is: 2 4 6 7 9 9
Index array is: 3 2 1 5 4 6

Following activities are selected: 3 2 5 6
4 activities are selected in total

...Program finished with exit code 0
Press ENTER to exit console.
```

2) Knapsack Algorithm (Code and Output):

```
#include<stdio.h>
void main()
{
   int n,i,j,t,C,s;
   printf("\n Enter the number of vegetables: ");
   scanf("%d",&n);
   printf("\n Enter the capacity of the bag: ");
   scanf("%d",&C);
   int pt1[n],wt1[n];
   printf("\n Enter the profit array: ");
   for(i=0;i<n;i++)
   scanf("%d",&pt1[i]);
   printf("\n Enter the weight array: ");
   for(i=0;i<n;i++)
   scanf("%d",&wt1[i]);
```

```
int pt2[n],pt3[n],wt2[n],wt3[n];
for(i=0;i<n;i++)
{
   pt2[i]=pt1[i];
   pt3[i]=pt1[i];
   wt2[i]=wt1[i];
   wt3[i]=wt1[i];
}
printf("\n ***Approach: Maximum Profit***\n");
for(i=0;i<n;i++)
{
   for(j=i+1;j<n;j++)
   {
      if(pt1[i]<pt1[j])
      {
         t=pt1[i];
         pt1[i]=pt1[j];
         pt1[j]=t;
         t=wt1[i];
         wt1[i]=wt1[j];
         wt1[j]=t;
      }
   }
}
printf("\n Sorted arrays are: ");
printf("\n Profit array: ");
for(i=0;i<n;i++)
printf("%d ",pt1[i]);
printf("\n Weight array: ");
for(i=0;i<n;i++)
printf("%d ",wt1[i]);
s=same(wt1,pt1,n,C);

printf("\n\n ***Approach: Minimum Weight***\n");
for(i=0;i<n;i++)
{
   for(j=i+1;j<n;j++)
   {
      if(wt2[i]>wt2[j])
```

```
        {
            t=wt2[i];
            wt2[i]=wt2[j];
            wt2[j]=t;
            t=pt2[i];
            pt2[i]=pt2[j];
            pt2[j]=t;
        }
    }
}
printf("\n Sorted arrays are: ");
printf("\n Profit array: ");
for(i=0;i<n;i++)
printf("%d ",pt2[i]);
printf("\n Weight array: ");
for(i=0;i<n;i++)
printf("%d ",wt2[i]);
s=same(wt2,pt2,n,C);

printf("\n\n ***Approach: Maximum Density***\n");
float dt[n];
for(i=0;i<n;i++)
{
    dt[i]=pt3[i]/wt3[i];
}
 for(i=0;i<n;i++)
{
    for(j=i+1;j<n;j++)
    {
        if(dt[i]<dt[j])
        {
            t=dt[i];
            dt[i]=dt[j];
            dt[j]=t;
            t=pt3[i];
            pt3[i]=pt3[j];
            pt3[j]=t;
            t=wt3[i];
            wt3[i]=wt3[j];
            wt3[j]=t;
```

```
            }
        }
    }
    printf("\n Sorted arrays are: ");
    printf("\n Density array: ");
    for(i=0;i<n;i++)
    printf("%f ",dt[i]);
    printf("\n Profit array: ");
    for(i=0;i<n;i++)
    printf("%d ",pt3[i]);
    printf("\n Weight array: ");
    for(i=0;i<n;i++)
    printf("%d ",wt3[i]);
    s=same(wt3,pt3,n,C);
}
void same(int wt[],int pt[],int n,int C)
{
    int i,j;
    for(i=0;i<n;i++)
    {
        if(wt[i]<=C)
        {
            C=C-wt[i];
            if(C==0)
            break;
        }
        else
        {
            pt[i]=(pt[i]*C)/wt[i];
            C=0;
            break;
        }
    }
    int sum=0;
    for(j=0;j<=i;j++)
    {
        sum=sum+pt[j];
    }
    printf("\n\n Maximum profit: %d",sum);
}
```

```
Enter the number of vegetables: 7

Enter the capacity of the bag: 10

Enter the profit array: 5 10 15 7 8 9 4

Enter the weight array: 1 3 5 4 1 3 2

***Approach: Maximum Profit***

Sorted arrays are:
Profit array: 15 10 9 8 7 5 4
Weight array: 5 3 3 1 4 1 2

Maximum profit: 31

***Approach: Minimum Weight***

Sorted arrays are:
Profit array: 5 8 4 9 10 1 15
Weight array: 1 1 2 3 3 4 5

Maximum profit: 36

***Approach: Maximum Density***

Sorted arrays are:
Density array: 8.000000 5.000000 3.000000 3.000000 3.000000 2.000000 1.000000
Profit array: 8 5 15 10 9 4 7
Weight array: 1 1 5 3 3 2 4

Maximum profit: 38

...Program finished with exit code 0
Press ENTER to exit console.
```

Conclusion: We have successfully implemented Activity Selection algorithm and Knapsack algorithm using C.