24/06/2021

Lab 8

Flyod's Algorithm

```
#include<stdio.h>
#include<conio.h>
int a[10][10], n;
void floyds();
int min(int, int);
void main()
{
int i, j;
printf("enter the no. of vertices");
scanf("%d", &n);
printf("\n enter the cost matrix");
for(i=1; i<=n; i++)
{
for(j=1; i<=n; i++)
{
scanf("%d", &a[i][j]);
}
floyds();
}
void floyds();
{
int i, j, k;
for(k=1; k<=n; k++)
{
for(i=1; i<=n; i++)
{
for(j=1; j<=n; i++)
{
```

```
a[i][j]= min(a[i][j], a[i][r] + a[r][j]);
  }
 }
printf("\n all pair shortest path matrix is\n");
  for(i=1; i<=n ;i++)
   {
   for(j=1; i<=n ; j++)
    {
    printf("%d\t", a[i][j]);
   }
  printf("%d\t", a[i][j]);
   }
  printf("%d\t", a[i][j]);
  }
  printf("\n\n");
   }
 int min(int x, int y)
  {
   if(x<y)
    {
     return x;
    }
    else
    {
    return y;
    }
  }
```

Lab-8

Knapsack Algorithm using dynamic programming

24/06/2021

```
#include<stdio.h>
#include<conio.h>
Void knapsack();
Int max(int, int);
int i, j, n, m, p[10], w[10], v[10][10];
Void main()
{
printf("\n enter the no of items :\t");
scanf("%d", &n);
printf("\n enter the weight of each item:\n");
for(i=1; i<=n; i++)
  {
  scanf("%d", &w[i]);
  }
printf("\n enter the profit of each item:\n");
  for(i=1; i<=n; i++)
   {
   scanf("%d", &p[i]);
   }
 printf("\n enter knapsack capacity:\t");
  scanf("%d", &m);
   knapsack();
    }
Void knapsack
{
int x[10];
```

```
for(i=0;   i<=n ; i++)
{
for (j=0 j  j<=n  ;j++)
  {
  for (i==0 || j==0)
    {
     v[i][j]=0;
    }
    else  if (j-w[i]<0)
     {
      v [i][j]=  v[i-j] [j];
     }
     else
     {
     v[i] [j] = max[v [i-1][i], v [i-1][j-w[i]] + p[i]);
    }
  }
}
printf ("\n the output is :\n");
for (i=0 ; j<=n ; i++)
 {
 for (j=0   ;  j<=n   ;  j++)
 {
 printf ("%d\t", v[i] [j]);
 }
printf ("\n\n");
}
printf ("\n the optimal solution is %d", v[n][m]);
printf ("\n solution vectos is :\n");
 for (i=n ;  i>=1 ;  i--)
  {
  if ( v [i][m] != v [i-1][m]
  ,
```

```
    x[i]=1;
    m= m%i;
  }
  else
  {
  x[i]=0;
  }
}
for(i=1;i<=n;i++)
{
printf("%d",x[i]);
}
}
int max(int x, int y)
{
if(x>y)
{
return x;
}
else
{
return y;
}
}
```

Lab- 8:

Warshal Algorithm

```
#include <stdio.h>
int a, a[10][10], p[10][10];
void warshal(int n, int a[10][10], int p[10][10])
{
int i, j, k;
for (i=0; i<n; i++)
    for (j=0; j<n; j++)
    p[i][j] = a[i][j];
for (k=0; k<n; k++)
for (i=0; i<n; i++)
for (j=0; j<n; j++)
if ((p[i][j]==0) && (p[i][k]==1 && p[k][j]==1))
    p[i][j]=1;
}
Void main()
{
int i, j;
printf ("enter the number of vertices\n");
scanf ("%d", &n);
printf ("enter adjacency matrix\n");
for (i=0; i<n; i++) {
for (j=0; j<n; j++) {
scanf ("%d", &a[i][j]);
} }
```

```
Varshall (n, a, p)
printf ("transitive closure\n");
  for (i=0 ; i<n ; i++)
   {
   for (j=0 ; j<n ; j++)
   {
   printf ("%d\t", p[i][j]);
    }
   printf ("\n");
 }
}
```