**Assignment No:-**

**Assignment Name:- Implementation of program for solution of Greedy Knapsack.**

**Name:- WARKE PURVA DILIP.**

**Roll No:- 136.**

#include<iostream.h>

#include<conio.h>

//Solution for Greedy knapsack

class GREEDY

{

private:

int n;

float M,Cu,\*P,\*W,\*X,\*R;

public:

GREEDY(float size, int par); //size=Maximum capacity

void READ(); //to read data from user

void SORT();

void KNAPSACK();

void DISPLAY();

};

GREEDY :: GREEDY(float size,int par)

{

M=size;

n=par;

P=new float[n+1];//for profit

W=new float[n+1];//for weight

X=new float[n+1];//for soln vector

R=new float[n+1];//for Ratio p/w

}

void GREEDY :: READ()

{

for(int i=1; i<=n; i++) //Data read by user

{

cout<<"Enter weight of product"<<i<<" : ";

cin>>W[i];

cout<<"Enter Profit of Product"<<i<<" : ";

cin>>P[i];

}

}

void GREEDY :: SORT() //sort according to P/W in form ratio and sort that ratio

{

float temp;

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

{

R[i]=P[i]/W[i]; // p/w ratio stored in a R array

}

//sorting ratio of p/w

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

{

for(int j=1;j<=n-i;j++)

{

if(R[j] < R[j+1])

{

temp=R[j];

R[j]=R[j+1];

R[j+1]=temp;

temp=P[j];

P[j]=P[j+1];

P[j+1]=temp;

temp=W[j];

W[j]=W[j+1];

W[j+1]=temp;

}

}

}

}

void GREEDY :: KNAPSACK()

{

Cu = M; //remaining Knapsack capacity

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

{

if(W[i]> Cu)

break;

else

{

X[i] =1;

Cu= Cu-W[i];

}

}

if(i<=n)

{

X[i]= Cu/W[i];

}

}

void GREEDY :: DISPLAY()

{

float sum=0;

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

{

sum= sum + (X[i]\*P[i]);

}

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

{

cout<<" \n Weight: "<<W[i]<<" Profit: "<<P[i]<<" Ratio "<<R[i]<<endl;

cout<<"Solution vector is: "<<X[i]<<endl<<endl; //solution vector

}

//Maximum profit of products

cout<<"\nMax Profit is: "<<sum<<endl;

}

void main()

{

clrscr();

int n;

float size;

cout<<"Enter Capacity of Knapsack Bag: ";

cin>>size;

cout<<"Enter No. of Products: ";

cin>>n;

GREEDY gk(size,n);

gk.READ();

gk.SORT();

gk.KNAPSACK();

gk.DISPLAY();

getch();

}