```cpp
//*****************************************************************//
//*************string  Author = "SACHIN SAINI " *****************//
//*****************************************************************//


// Time Complexity: O(nlogn) for sorting and O(n) for selecting process
#include<iostream>
using namespace std;

struct items //Structure that contains the weight, price and calculated ratio.
{
    int weight;
    int price;
    float ratio;
}temp; //temp variable taken for swap two structure
int fractionalKnapsack(struct item arr[],int n,int W); //Fractional knapsack
int partition(struct items arr[],int left,int right);  //Partition function of quick sort
void quickSort(struct items arr[],int left,int right); //quick sort

int main()
{
    struct items arr[10]; //create a array of structure
    int n;
    cout<<"Enter the total number of items \n";
    cin>>n;
    for(int i=0;i<n;i++)//take input from users
    {
        cout<<"Enter weight\n";
        cin>>arr[i].weight;
        cout<<"Enter price \n";
        cin>>arr[i].price;
        arr[i].ratio=arr[i].price/arr[i].weight; //ratio is calculated using divide price by weight.
    }

    quickSort(arr,0,n-1); //first sort the items according to ratio

    for(int i=0;i<n;i++) //for printing the structure
    {
        cout<<arr[i].weight<<" "<<arr[i].price<<endl;
    }
    float MaximumPriceValue=fractionalKnapsack(arr,n,60); //now use fractional knapsack to find the maximum price things can be put into the bag
    cout<<MaximumPriceValue<<endl;
}
////////////////////////////////Quick Sort on structure //////////////////////////
int partition(struct items arr[],int left,int right)
{
    int i=left;
    int pivot=arr[left].ratio;
    for(int j=left+1;j<=right;j++)
    {
        if(arr[j].ratio>pivot)
        {
            i++;
            temp=arr[i];
            arr[i]=arr[j];
            arr[j]=temp;
        }
    }
    temp=arr[left];
    arr[left]=arr[i];
    arr[i]=temp;
    return i;
}
void quickSort(struct items arr[],int left,int right)
{
    if(left<=right)
    {
```

```
68          int middle=partition(arr,left,right);
69          quickSort(arr,left,middle-1);
70          quickSort(arr,middle+1,right);
71      }
72  }
73  // ////////////////////////////////////////////////////////////////////////////
74
75  int fractionalKnapsack(struct items arr[],int n,int W)
76  {
77      int curr_weight=0;
78      float final_profit=0,remaining=0;
79      for(int i=0;i<n;i++)
80      {
81          //current weight + item weight which is to be put into the bag then simply put
            it.
82          if(curr_weight+arr[i].weight<=W)
83          {
84              curr_weight+=arr[i].weight;
85              final_profit+=arr[i].price;
86              cout<<" Final_profit = "<<final_profit<<" ";
87          }
88          else //other-wise take fractional part of item and calculate the final_price
89          {
90              remaining=W-curr_weight;//remaining bag capacity
91              cout<<"Remaining="<<remaining<<" arr[i].weight= "<<arr[i].weight<<"
                arr[i].price="<<arr[i].price<<endl;
92              float a=(float)(remaining/arr[i].weight)*arr[i].price; //calculate the
                fraction part price
93              cout<<"A="<<a<<"\n";
94              final_profit+=a;  //add that fraction part price to the final_profit
95              cout<<" Final_profit = "<<final_profit<<" ";
96              break; //no need to do further so break;
97          }
98      }
99      return final_profit;
100 }
101
```