AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Science and Technology

Mid-Term Assignment

Submitted by-

Name- Ruksat Khan Shayoni

ID- 20-41922-1

Course- Algorithms

Section- L

Semester- Spring 20-21

Submitted to-

Dr. Md. Asraf Ali

Associate Professor,

Department of CSE,

FST, AIUB

Write a code for preparing a note on fractional knapsack in order to teach your juniors

Code:

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct Item {
char ID[5];
int weight;
int value;
float density;
};
void fractionalKnapsack(Item items[], int n, int w);



int main() {
int i, j;

Item items[6] = {
{"i1", 2, 2, 0},
{"i2", 8, 5, 0},
{"i3", 4, 5, 0},
{"i4", 8, 3, 0},
{"i5", 4, 9, 0},
{"i6", 2, 6, 0}
};
int n = 6;
cout<<"\n\n\t\t\t\t\t\tKnapsack with highest weight capacity: 15 "<<endl;
cout<<"\t\t\t\t\t\tWe will have to fill the KNAPSACK with Items such that the benefit is the Maximum:-"<<endl;

cout<<"\n\t\t\t\t\t\t=========================================================================="<<endl;
```

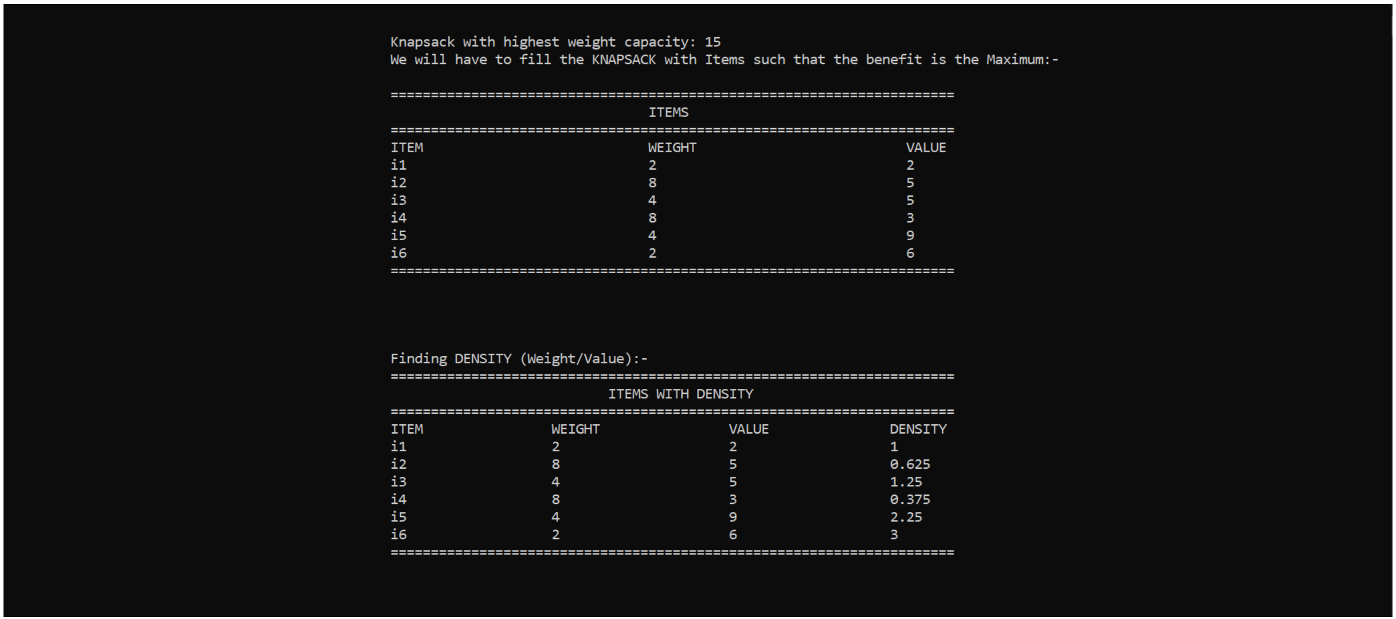
```
cout<<"\t\t\t\t\t\t\t\t\t       ITEMS       "<<endl;
cout<<"\t\t\t\t\t\t====================================================
"<<endl;
cout<<"\t\t\t\t\t\tITEM"<<"\t\t\t\tWEIGHT"<<"\t\t\t\tVALUE"<<endl;
for(int q=0;q<n;q++)
cout<<"\t\t\t\t\t\t"<<items[q].ID<<"\t\t\t\t"<<items[q].weight<<"\t\t\t\t"<<items[q].value<<"
"<<endl;
cout<<"\t\t\t\t\t\t====================================================
"<<endl;



Item temp;



int W = 15;


//compute desity = (value/weight)
for(i = 0; i < n; i++) {
items[i].density = float(items[i].value) / items[i].weight;
}
cout<<"\n\n\n\n\t\t\t\t\t\tFinding DENSITY (Weight/Value):-";


cout<<"\n\t\t\t\t\t\t==================================================
=="<<endl;
cout<<"\t\t\t\t\t\t\t\t\t   ITEMS WITH DENSITY       "<<endl;
cout<<"\t\t\t\t\t\t====================================================
"<<endl;
cout<<"\t\t\t\t\t\tITEM"<<"\t\t    WEIGHT"<<"\t\t  VALUE"<<"\t\t     DENSITY"<<endl;


for(int q=0;q<n;q++)
cout<<"\t\t\t\t\t\t"<<items[q].ID<<"\t\t    "<<items[q].weight<<"\t\t\t  "<<items[q].value<<"\t\t
"<<items[q].density<<endl;
cout<<"\t\t\t\t\t\t====================================================
"<<endl;
//sort by density in descending order
```

```
for(i = 1; i < n; i++) {
for(j = 0; j < n - i; j++) {
if(items[j+1].density > items[j].density)
{
temp = items[j];
items[j] = items[j+1];
items[j+1] = temp;
}
}
}
cout<<"\n\n\n\n\t\t\t\t\t\tSorting the table as per value of DENSITY:-";

cout<<"\n\t\t\t\t\t\t================================================================================="<<endl;
cout<<"\t\t\t\t\t\t\t\t     ITEMS AFTER SORTING AS PER DENSITY"<<endl;
cout<<"\t\t\t\t\t\t================================================================================="<<endl;

cout<<"\t\t\t\t\t\tITEM"<<"\t\t    WEIGHT"<<"\t\t  VALUE"<<"\t\t      DENSITY"<<endl;
for(int q=0;q<n;q++)
cout<<"\t\t\t\t\t\t"<<items[q].ID<<"\t\t    "<<items[q].weight<<"\t\t\t  "<<items[q].value<<"\t\t "<<items[q].density<<endl;
cout<<"\t\t\t\t\t\t================================================================================="<<endl;
cout<<"\n\n\n\n\t\t\t\t\t\tKnapsack Calculation:-";

cout<<"\n\n\t\t\t\t\t\t================================================================================"<<endl;
cout<<"\t\t\t\t\t\t\t\t\t   KNAPSACK CALCULATION"<<endl;
cout<<"\t\t\t\t\t\t================================================================================="<<endl;
cout<<"\t\t\t\t\t\tITEM"<<"\tWEIGHT"<<"\t\tVALUE"<<"\t\tTOTAL WEIGHT"<<"\tTOTAL BENIFIT"<<endl;

fractionalKnapsack(items, n, W);
```

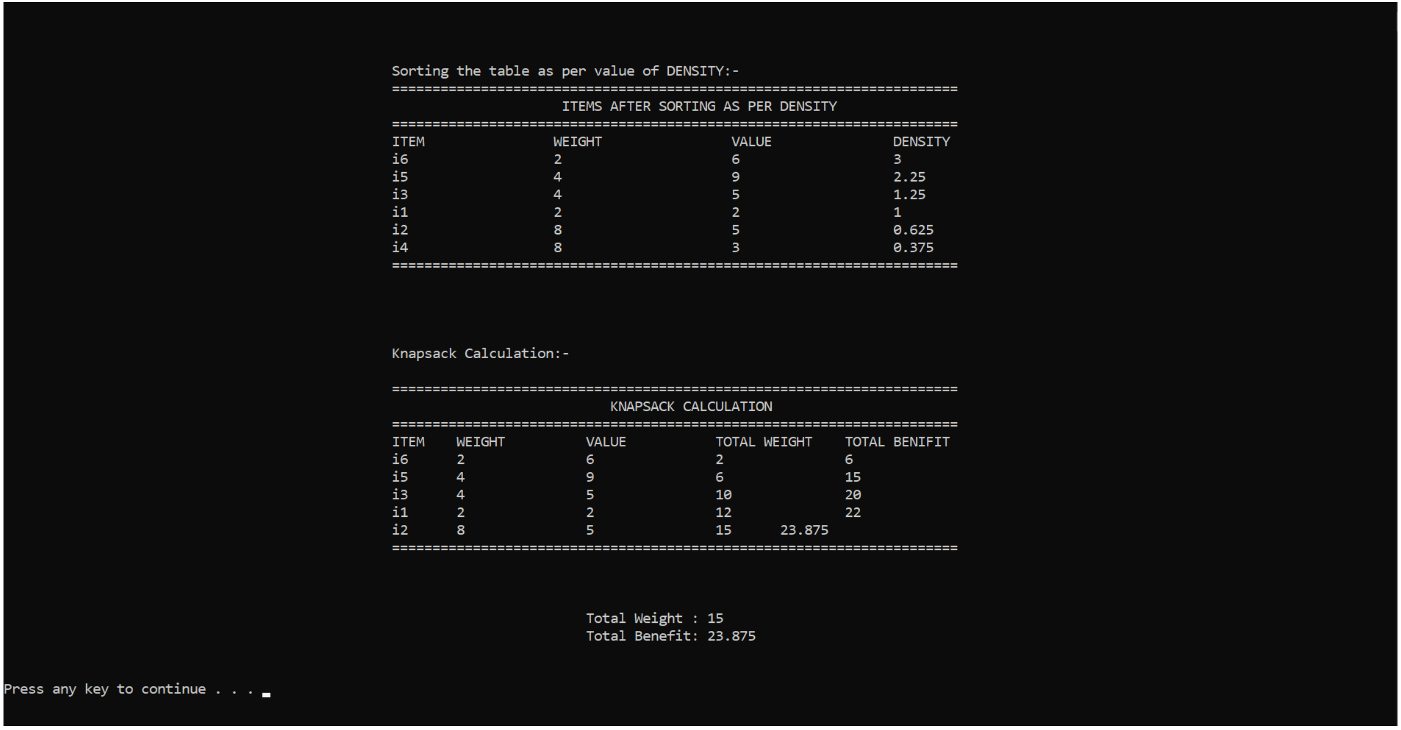
```
return 0;
}
void fractionalKnapsack(Item items[], int n, int W) {
int i, wt;
float value;
float totalWeight = 0, totalBenefit = 0;
for(i = 0; i < n; i++) {
if(items[i].weight + totalWeight <= W) {
totalWeight += items[i].weight;
totalBenefit += items[i].value;

cout<<"\t\t\t\t\t\t"<<items[i].ID<<"\t"<<items[i].weight<<"\t\t"<<items[i].value<<"\t\t"<<totalWeight<<"\t\t"<<totalBenefit<<endl;

}
else {
wt = (W - totalWeight);
value = wt * (float(items[i].value) / items[i].weight);
totalWeight += wt;
totalBenefit += value;
cout<<"\t\t\t\t\t\t"<<items[i].ID<<"\t"<<items[i].weight<<"\t\t"<<items[i].value<<"\t\t"<<totalWeight<<"\t"<<totalBenefit<<endl;
break;
}
}
cout<<"\t\t\t\t\t\t==================================================================="<<endl;
cout<<"\n\n\n\t\t\t\t\t\t\t\t\tTotal Weight : "<<totalWeight<<endl;
cout<<"\t\t\t\t\t\t\t\t\tTotal Benefit: "<<totalBenefit<<"\n\n"<<endl;

system("pause");
}
```

-----------------------------X-----------------------------

Output:



```
Knapsack with highest weight capacity: 15
We will have to fill the KNAPSACK with Items such that the benefit is the Maximum:-

======================================================================
                                ITEMS
======================================================================
ITEM                            WEIGHT                          VALUE
i1                              2                               2
i2                              8                               5
i3                              4                               5
i4                              8                               3
i5                              4                               9
i6                              2                               6
======================================================================



Finding DENSITY (Weight/Value):-
======================================================================
                           ITEMS WITH DENSITY
======================================================================
ITEM                WEIGHT                VALUE               DENSITY
i1                  2                     2                   1
i2                  8                     5                   0.625
i3                  4                     5                   1.25
i4                  8                     3                   0.375
i5                  4                     9                   2.25
i6                  2                     6                   3
======================================================================
```



```
                                                Sorting the table as per value of DENSITY:-
                                                ======================================================================
                                                                     ITEMS AFTER SORTING AS PER DENSITY
                                                ======================================================================
                                                ITEM                WEIGHT                VALUE               DENSITY
                                                i6                  2                     6                   3
                                                i5                  4                     9                   2.25
                                                i3                  4                     5                   1.25
                                                i1                  2                     2                   1
                                                i2                  8                     5                   0.625
                                                i4                  8                     3                   0.375
                                                ======================================================================



                                                Knapsack Calculation:-

                                                ======================================================================
                                                                           KNAPSACK CALCULATION
                                                ======================================================================
                                                ITEM    WEIGHT          VALUE           TOTAL WEIGHT    TOTAL BENIFIT
                                                i6      2               6               2               6
                                                i5      4               9               6               15
                                                i3      4               5               10              20
                                                i1      2               2               12              22
                                                i2      8               5               15      23.875
                                                ======================================================================


                                                                        Total Weight : 15
                                                                        Total Benefit: 23.875

Press any key to continue . . .
```

-----------The End----------