**package** GreedyAlgorithm;

**import** java.util.ArrayList;

**import** java.util.Collections;

**class** ItemsWithPriceWeight **implements** Comparable<ItemsWithPriceWeight>

{

**public int itemName**;

**public int profit**;

**public int weight**;

**public** ItemsWithPriceWeight(**int** itemName, **int** profit, **int** weight) {

**this**.**itemName** = itemName;

**this**.**profit** = profit;

**this**.**weight** = weight;

}

@Override

**public int** compareTo(ItemsWithPriceWeight o) {

**if** (**this**.**profit**<o.**profit**) **return** 1;

**else return** -1;

}

}

**class** GetItemWithProfit

{

ArrayList<ItemsWithPriceWeight> **arrayList**;

**public double**[] **calculateProfit**;

**public int maxWeightCapacity**;

**public int indexForProfitCalculate**;

**public int**[] **object**;

**public double totalProfit**;

**public** GetItemWithProfit(ArrayList<ItemsWithPriceWeight> arrayList, **double**[] calculateProfit, **int** maxWeightCapacity) {

**this**.**arrayList** = arrayList;

**this**.**calculateProfit** = calculateProfit;

**this**.**maxWeightCapacity** = maxWeightCapacity;

**this**.**indexForProfitCalculate** = 0;

**this**.**object** = **new int**[100];

**this**.**totalProfit** = 0;

}

**public void** giveItemWithMaxProfit()

{

**for** (ItemsWithPriceWeight print : **arrayList**)

{

takeItemWithMaxProfit(print.**itemName**,print.**profit**,print.**weight**);

}

}

**public void** takeItemWithMaxProfit(**int** itemName, **int** profit, **int** weight)

{

**maxWeightCapacity** = **maxWeightCapacity** - weight;

**if** (**maxWeightCapacity** ==0) **return**;

**else if** (**maxWeightCapacity**<0)

{

**maxWeightCapacity** = **maxWeightCapacity** + weight;

**double** profitFraction = (Double.*valueOf*(**maxWeightCapacity**)/Double.*valueOf*(weight));

**double** take = profitFraction \* Double.*valueOf*(profit);

**calculateProfit**[**indexForProfitCalculate**] = take;

**object**[**indexForProfitCalculate**] = itemName;

**indexForProfitCalculate**++;

**maxWeightCapacity** =0;

**return**;

}**else**

{

**calculateProfit**[**indexForProfitCalculate**] = profit;

**object**[**indexForProfitCalculate**] = itemName;

**indexForProfitCalculate** ++;

}

}

**public void** printTotalProfit()

{

**for** (**int** i = 0; i <**indexForProfitCalculate** ; i++) {

**if** (**calculateProfit**[i] ==0){}**else** {

**totalProfit** = **totalProfit** + **calculateProfit**[i];

System.***out***.println(**"item Name: "** + **object**[i] + **", profit is = "** + **calculateProfit**[i]);

}

}

System.***out***.println(**"----------------------------------"**);

System.***out***.println(**"Total profit if we choose max profit = "**+**totalProfit**);

}

}

**public class** FractionalKnapsackMaxProfit {

**public static void** main(String[] args) {

ArrayList<ItemsWithPriceWeight> object = **new** ArrayList<>();

object.add(**new** ItemsWithPriceWeight(1,5,1));

object.add(**new** ItemsWithPriceWeight(2,10,3));

object.add(**new** ItemsWithPriceWeight(3,15,5));

object.add(**new** ItemsWithPriceWeight(4,7,4));

object.add(**new** ItemsWithPriceWeight(5,8,1));

object.add(**new** ItemsWithPriceWeight(6,9,3));

object.add(**new** ItemsWithPriceWeight(7,4,2));

Collections.*sort*(object);

*/\* to calculate overall profit and max weight \*/*

**double**[] profit = **new double**[100];

**int** maxWeightCapacity = 15;

GetItemWithProfit object1 = **new** GetItemWithProfit(object,profit,maxWeightCapacity);

object1.giveItemWithMaxProfit();

object1.printTotalProfit();

}

}