

|  |
| --- |
| **Title: Implementation of Knapsack Problem** |



**Objective:** To learn the Greedy strategy of solving the problems for different types of problems

**CO to be achieved:**

|  |  |
| --- | --- |
| Sr. No | Objective |
| CO 1 | Compare and demonstrate the efficiency of algorithms using asymptotic complexity notations. |
| CO 2 | Analyze and solve problems for divide and conquer strategy, greedy method, dynamic programming approach and backtracking and branch & bound policies. |
| CO 3 | Analyze and solve problems for   different string matching algorithms. |



**Books/ Journals/ Websites referred:**

1. **Ellis horowitz, Sarataj Sahni, S.Rajsekaran,” Fundamentals of computer algorithm”, University Press**
2. **T.H.Cormen ,C.E.Leiserson,R.L.Rivest and C.Stein,” Introduction to algortihtms”,2nd Edition ,MIT press/McGraw Hill,2001**
3. **http://lcm.csa.iisc.ernet.in/dsa/node184.htm**
4. **http://students.ceid.upatras.gr/~papagel/project/kruskal.htm**
5. [**http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/kruskalAlgor.html**](http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/kruskalAlgor.html)
6. **http://lcm.csa.iisc.ernet.in/dsa/node183.html**
7. **http://students.ceid.upatras.gr/~papagel/project/prim.htm**
8. **http://www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf**



**Pre Lab/ Prior Concepts:**

Data structures, Concepts of algorithm analysis



**Historical Profile:**

The knapsack problem represents constraint satisfaction optimization problems’ family. Based on nature of constraints, the knapsack problem can be solved with various problem saolving strategies. Typically, these problems represent resource optimization solution.

Given a set of n inputs. · Find a subset, called feasible solution, of the n inputs subject to some constraints, and satisfying a given objective function. · If the objective function is maximized or minimized, the feasible solution is optimal. · It is a locally optimal method.

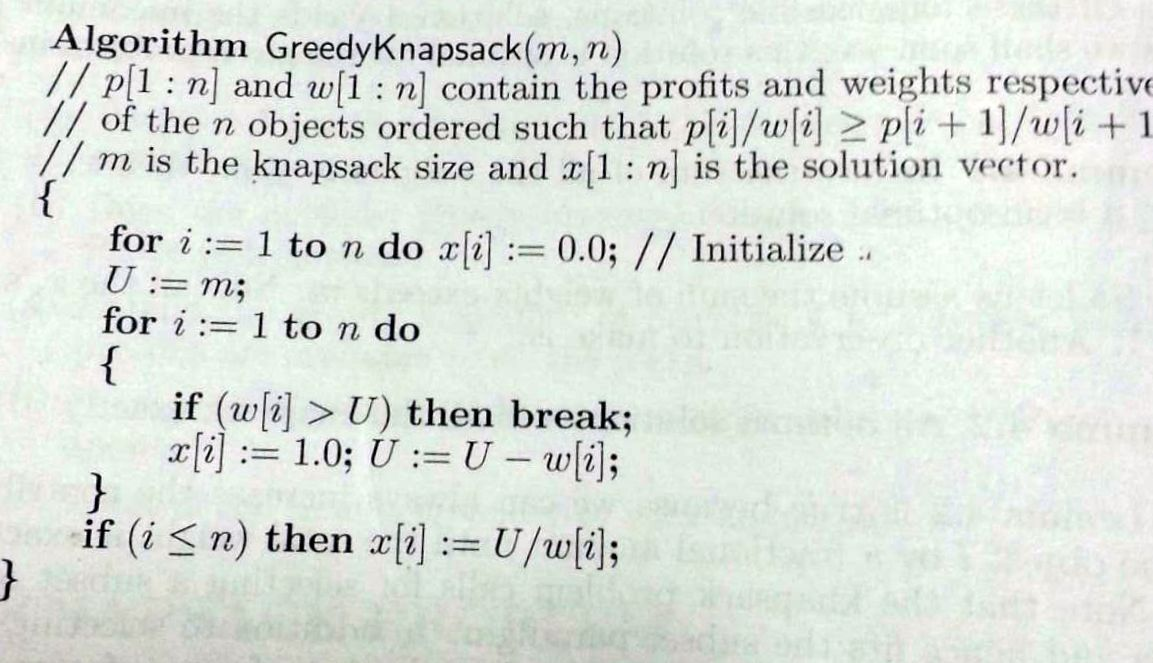


**New Concepts to be learned:**

Application of algorithmic design strategy to any problem, Greedy method of problem solving Vs other methods of problem solving, optimality of the solution, knapsack problem and their applications



**Knapsack Problem Algorithm**



**Example: Knapsack Problem**

Program:

import java.util.Scanner;

public class fractional\_knapsack{

public static void main(String[] args)

{

int n,m;

Scanner sc=new Scanner(System.in);

System.out.println("Enter total number of items: ");

n=sc.nextInt();

System.out.println("Enter capacity: ");

m=sc.nextInt();

System.out.println("Enter weights: ");

float w[]=new float[n];

for(int i=0;i<n;i++)

{

w[i]=sc.nextFloat();

}

System.out.println("Enter profits: ");

float p[]=new float[n];

for(int i=0;i<n;i++)

{

p[i]=sc.nextFloat();

}

ks(w,p,m,n);

}

public static void knapsack(float w[],float p[],int m,int n)

{

int i;

float pw[]=new float[n];

int q[]=new int[n];

for( i=0;i<n;i++)

{

pw[i]=p[i]/w[i];

q[i]=i+1;

}

for( i=0;i<n;i++)

{

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

{

if(pw[j+1]>pw[j])

{

float t=pw[j];

pw[j]=pw[j+1];

pw[j+1]=t;

float t1=w[j];

w[j]=w[j+1];

w[j+1]=t1;

float t2=p[j];

p[j]=p[j+1];

p[j+1]=t2;

int t3=q[i];

q[i]=q[i+1];

q[i+1]=t3;

}

}

}

float x[]=new float[n];

for(i=0;i<n;i++)

{

x[i]=0;

}

float U=m;

float profit=0;

for( i=0;i<n;i++)

{

if(w[i]>U)

{

x[i]=U/w[i];

profit+=p[i]\*x[i];

U=0;

break;

}

else

{

x[i]=1;

profit+=p[i];

U=U-w[i];

}

}

System.out.println();

System.out.println("Item no: \t Fraction: \t Profit: \tWeight: ");

for( i=0;i<n;i++)

{

System.out.println(q[i]+"\t\t"+x[i]+"\t\t"+p[i]+"\t\t"+w[i]);

}

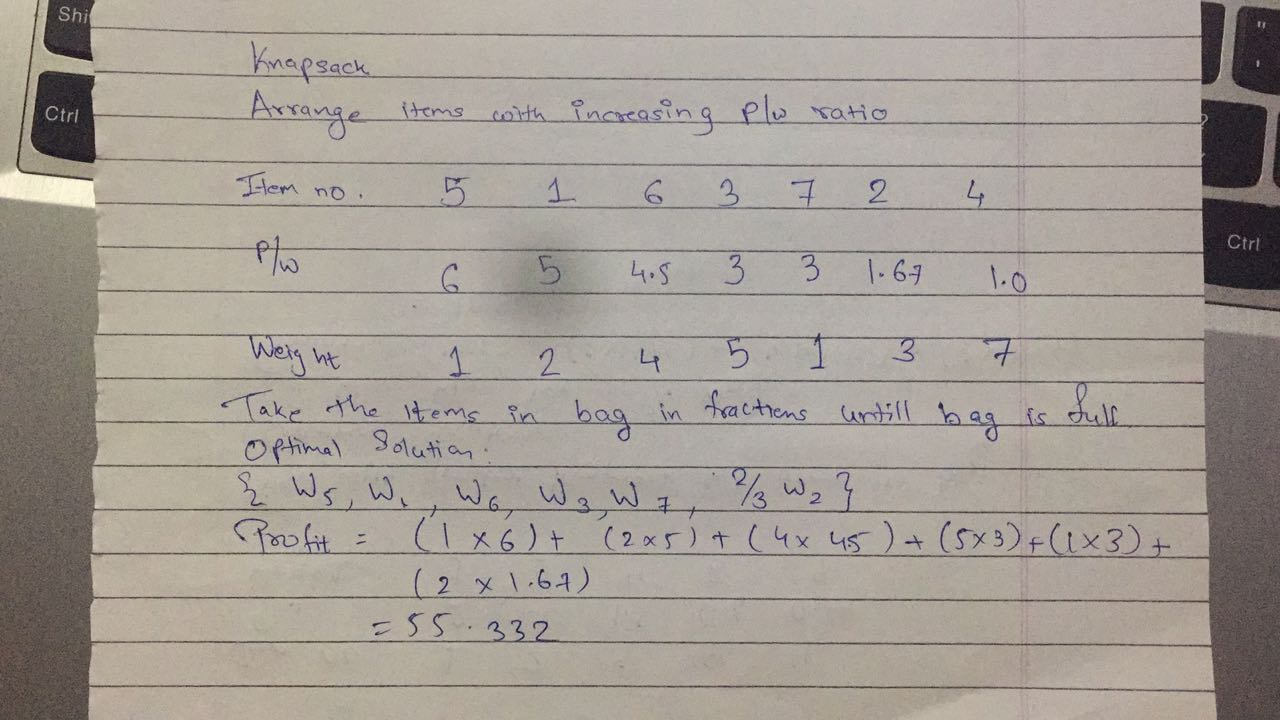
System.out.println("Maximum profit: "+profit);

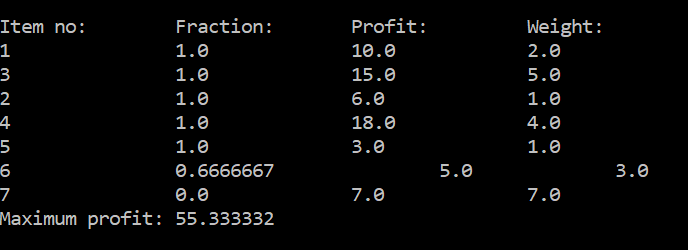
}

}

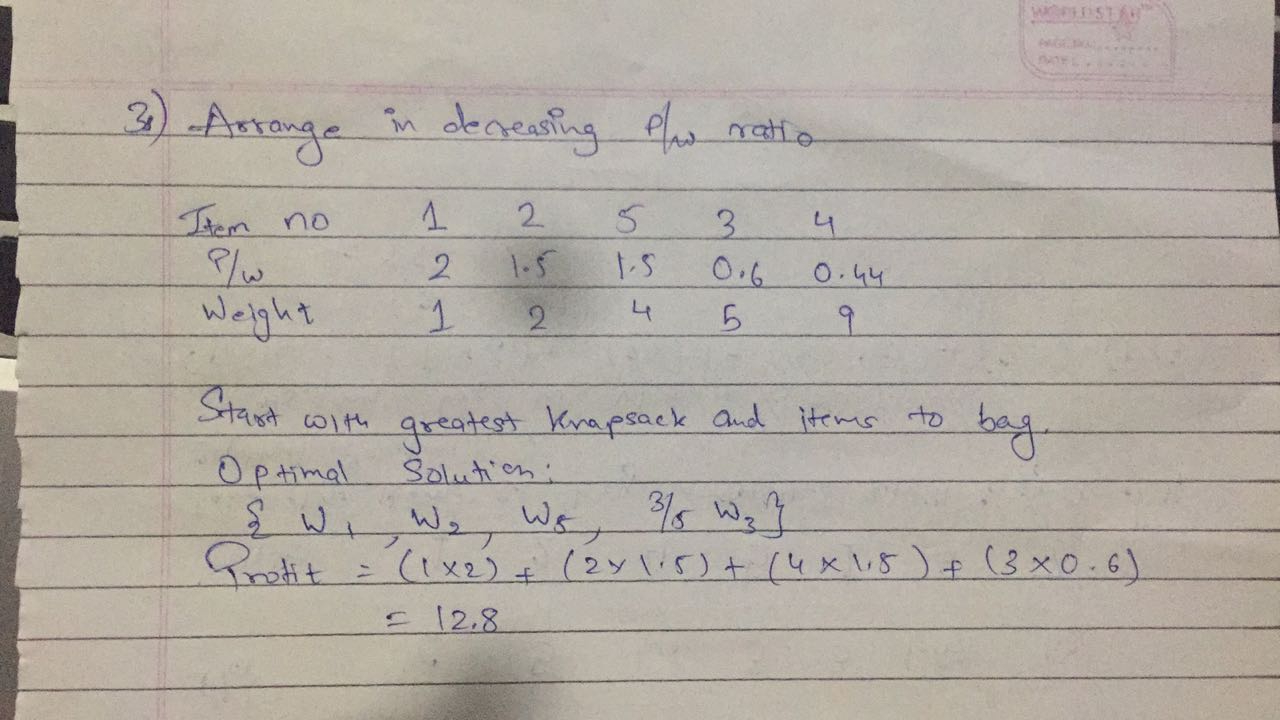
Output:

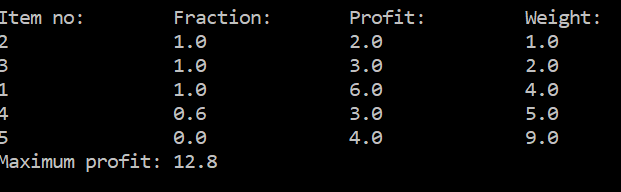
1)





2)





**Analysis of Knapsack Problem algorithm:**

**Time complexity:**

**If the provided items are already sorted into a decreasing order of piwi , then it takes a time in O(n);**

**Therefore, the total time including the sort is in O(n logn).**

**CONCLUSION:**

implementation fractional knapsack using greedy strategy successfully