****

**Southeast University**

**Department of Computer Science and Engineering (CSE)**

**School of Sciences and Engineering**

**Semester: (Spring, Year: 2025)**

**LAB REPORT NO**: **02**

**Course Title**: Algorithm Lab

**Course Code:** CSE266.14

**Batch**: 65

**Lab Experiment Name:** **Solving the Fractional Knapsack Problem Using Greedy Algorithm.**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | Md. Mahmud Hossain | 2023200000799 |

**Submission Date : 11-05-25**

**Course Teacher’s Name : Maisha Muntaha**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

**1) TITLE OF THE LAB REPORT EXPERIMENT:**

Solving the Fractional Knapsack Problem Using Greedy Algorithm

**2) OBJECTIVES:**

* To understand and implement the Greedy Algorithm approach for the Fractional Knapsack Problem.
* To sort items based on their value-to-weight ratio.
* To maximize the total profit within a given capacity limit using fractional inclusion of items.
* To reinforce the concept of algorithmic problem solving in real-world scenarios like resource allocation.

**3) PROCEDURE:**

1. Take the number of objects (n) as input from the user.
2. Accept the profit (value) and weight of each object.
3. Accept the maximum capacity of the knapsack (or drone in this context).
4. Calculate the value-to-weight ratio for each item.
5. Sort all items in descending order of their ratio using Insertion Sort.
6. Traverse through the sorted list:

* If the item fits entirely, add its full profit.
* If not, add a fractional amount based on remaining capacity.

1. Return and display the maximum profit obtained.

**4) IMPLEMENTATION:**

**Question:** You are managing a delivery drone for medical supplies in a remote area. The drone has a weight limit of 25 kg. You have the following medical packages available, each with a value (in importance) and a weight:

|  |  |  |
| --- | --- | --- |
| **Packages** | **Value** | **Weight** |
| Vaccine Kit | 90 | 10 |
| Emergency Fluids | 100 | 20 |
| Portable Oxygen | 120 | 30 |
| Pain Relief Kit | 60 | 10 |

* Write a C++ program that: Uses insertion sort to sort the packages by their value-to-weight ratio in descending order.
* Applies the Fractional Knapsack algorithm to choose packages that maximize total value without exceeding the drone’s capacity.

**Code:**

**= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =**

**#include<iostream>**

**using namespace std;**

**void insertionSort(int val[], int wt[], double Ratio[], int n)**

**{**

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

**{**

**double keyRatio =Ratio[i];**

**int keyVal = val[i];**

**int keyWt = wt[i];**

**int j = i - 1;**

**while (j >= 0 && Ratio[j] < keyRatio)**

**{**

**Ratio[j + 1] = Ratio[j];**

**val[j + 1] = val[j];**

**wt[j + 1] = wt[j];**

**j--;**

**}**

**Ratio[j + 1] = keyRatio;**

**val[j + 1] = keyVal;**

**wt[j + 1] = keyWt;**

**}**

**}**

**double fractionalKnapsack(int val[],int wt[],int n,int capacity)**

**{**

**double Ratio[n];**

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

**Ratio[i] = (double)val[i]/wt[i];**

**insertionSort(val,wt,Ratio,n);**

**cout<<"Sorted value-to-weight ratio: ";**

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

**{**

**cout<<Ratio[i]<<" ";**

**}**

**cout<<endl;**

**double res=0.0;**

**int currentCapacity = capacity;**

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

**{**

**if(wt[i]<=currentCapacity)**

**{**

**res+=val[i];**

**currentCapacity-=wt[i];**

**}**

**else**

**{**

**res+=((double)val[i]/wt[i]) \* currentCapacity;**

**break;**

**}**

**}**

**return res;**

**}**

**int main(){**

**int n;**

**cout<<"Enter total object: ";**

**cin>>n;**

**int val[n];**

**int wt[n];**

**int capacity;**

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

**{**

**cout<<"Enter profit of "<<i+1<<"th: ";**

**cin>>val[i];**

**cout<<"Enter weight of "<<i+1<<"th: ";**

**cin>>wt[i];**

**}**

**cout<<endl;**

**cout<<"Maximum Capacity of drone: ";**

**cin>>capacity;**

**cout<<endl;**

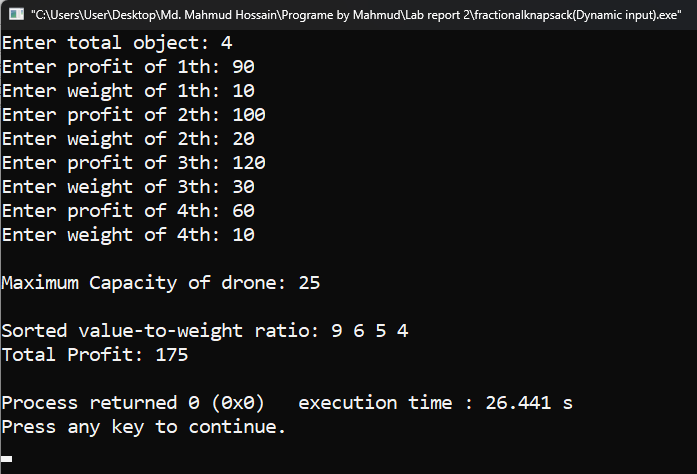
**double result = fractionalKnapsack(val,wt,n,capacity);**

**cout<<"Total Profit: "<<result<<endl;**

**return 0;**

**}**

**4) OUTPUT:**

****

**5) DISCUSSION:**

Finally, I got my result. In this lab task, I implemented the Fractional Knapsack problem using the Greedy Algorithm. I used Insertion Sort to sort items based on their value-to-weight ratio in descending order. The main challenge was managing three arrays (value, weight, and ratio) in sync during sorting, but I handled it carefully. I then filled the knapsack with the highest-ratio items, allowing fractional inclusion when necessary. This helped maximize the total profit. The program worked successfully and produced correct output, fulfilling the goal of optimizing resource usage within limited capacity.