7. Design and implement C/C++ Program to solve discrete Knapsack and continuous Knapsack

problems using greedy approximation method.

#include <stdio.h>

#define MAX 50

int p[MAX], w[MAX], x[MAX];

double maxprofit;

int n, m, i;

void greedyKnapsack(int n, int w[], int p[], int m)

{

double ratio[MAX];

// Calculate the ratio of profit to weight for each item

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

{

ratio[i] = (double)p[i] / w[i];

}

// Sort items based on the ratio in non-increasing order

for (i = 0; i < n - 1; i++)

{

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

{

if (ratio[i] < ratio[j])

{

double temp = ratio[i];

ratio[i] = ratio[j];

ratio[j] = temp;

int temp2 = w[i];

w[i] = w[j];

w[j] = temp2;

temp2 = p[i];

p[i] = p[j];

p[j] = temp2;

}

}

}

int currentWeight = 0;

maxprofit = 0.0;

// Fill the knapsack with items

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

{

if (currentWeight + w[i] <= m)

{

x[i] = 1; // Item i is selected

currentWeight += w[i];

maxprofit += p[i];

}

else

{

// Fractional part of item i is selected

x[i] = (m - currentWeight) / (double)w[i];

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

break;

}

}

printf("Optimal solution for greedy method: %.1f\n", maxprofit);

printf("Solution vector for greedy method: ");

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

printf("%d\t", x[i]);

}

int main()

{

printf("Enter the number of objects: ");

scanf("%d", &n);

printf("Enter the objects' weights: ");

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

scanf("%d", &w[i]);

printf("Enter the objects' profits: ");

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

scanf("%d", &p[i]);

printf("Enter the maximum capacity: ");

scanf("%d", &m);

greedyKnapsack(n, w, p, m);

return 0;

}

OUTPUT

Enter the number of objects: 4

Enter the objects' weights: 56 78 98 78

Enter the objects' profits: 23 45 76 78

Enter the maximum capacity: 100

Optimal solution for greedy method: 78.0

Solution vector for greedy method: 1 0 0 0