#include <stdio.h>

// Swap two integers

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

// Swap two floats

void swapFloat(float \*a, float \*b) {

float temp = \*a;

\*a = \*b;

\*b = temp;

}

// Sort by profit (descending order)

void sortByProfitDesc(int weights[], int values[], int n) {

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

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

if (values[j] > values[i]) {

swap(&values[i], &values[j]);

swap(&weights[i], &weights[j]);

}

}

}

}

// Sort by profit (ascending order)

void sortByProfitAsc(int weights[], int values[], int n) {

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

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

if (values[j] < values[i]) {

swap(&values[i], &values[j]);

swap(&weights[i], &weights[j]);

}

}

}

}

// Sort by profit/weight ratio (descending order)

void sortByRatioDesc(int weights[], int values[], int n) {

float ratios[n];

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

ratios[i] = (float)values[i] / weights[i];

}

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

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

if (ratios[j] > ratios[i]) {

swap(&values[i], &values[j]);

swap(&weights[i], &weights[j]);

swapFloat(&ratios[i], &ratios[j]);

}

}

}

}

// Fractional knapsack function

float fractionalKnapsack(int capacity, int weights[], int values[], int n) {

float totalValue = 0.0;

for (int i = 0; i < n && capacity > 0; i++) {

if (weights[i] <= capacity) {

capacity -= weights[i];

totalValue += values[i];

} else {

totalValue += values[i] \* ((float)capacity / weights[i]);

capacity = 0;

}

}

return totalValue;

}

// Print items

void printItems(int weights[], int values[], int n) {

printf("Items (weight, value): ");

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

printf("(%d, %d) ", weights[i], values[i]);

}

printf("\n");

}

int main() {

int n, capacity;

printf("Enter number of items: ");

scanf("%d", &n);

int weights[n], values[n];

int weightsCopy[n], valuesCopy[n];

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

printf("Item %d weight: ", i + 1);

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

printf("Item %d value: ", i + 1);

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

}

printf("Enter capacity of knapsack: ");

scanf("%d", &capacity);

// Method 1: More profit first

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

weightsCopy[i] = weights[i];

valuesCopy[i] = values[i];

}

sortByProfitDesc(weightsCopy, valuesCopy, n);

printf("\nMethod 1: More profit first\n");

printItems(weightsCopy, valuesCopy, n);

printf("Max value = %.2f\n", fractionalKnapsack(capacity, weightsCopy, valuesCopy, n));

// Method 2: Less profit first

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

weightsCopy[i] = weights[i];

valuesCopy[i] = values[i];

}

sortByProfitAsc(weightsCopy, valuesCopy, n);

printf("\nMethod 2: Less profit first\n");

printItems(weightsCopy, valuesCopy, n);

printf("Max value = %.2f\n", fractionalKnapsack(capacity, weightsCopy, valuesCopy, n));

// Method 3: Profit/weight ratio first

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

weightsCopy[i] = weights[i];

valuesCopy[i] = values[i];

}

sortByRatioDesc(weightsCopy, valuesCopy, n);

printf("\nMethod 3: Profit/weight ratio first\n");

printItems(weightsCopy, valuesCopy, n);

printf("Max value = %.2f\n", fractionalKnapsack(capacity, weightsCopy, valuesCopy, n));

return 0;

}

