**Fractional Knapsack**

#include <stdio.h>

#include <stdlib.h>

typedef struct item

{

  int weight;

  int profit;

  float x;

  int initialIndex;

} item;

void merge(float \*arr, item \*items, int l, int m, int r)

{

  int n1 = m - l + 1;

  int n2 = r - m;

  float \*L = (float \*)malloc(n1 \* sizeof(float));

  float \*R = (float \*)malloc(n2 \* sizeof(float));

  item \*Litems = (item \*)malloc(n1 \* sizeof(item));

  item \*Ritems = (item \*)malloc(n2 \* sizeof(item));

  int i, j, k;

  for (i = 0; i < n1; i++)

  {

    L[i] = arr[l + i];

    Litems[i] = items[l + i];

  }

  for (j = 0; j < n2; j++)

  {

    R[j] = arr[m + 1 + j];

    Ritems[j] = items[m + 1 + j];

  }

  i = 0;

  j = 0;

  k = l;

  while (i < n1 && j < n2)

  {

    if (L[i] >= R[j])

    {

      arr[k] = L[i];

      items[k] = Litems[i];

      i++;

    }

    else

    {

      arr[k] = R[j];

      items[k] = Ritems[j];

      j++;

    }

    k++;

  }

  while (i < n1)

  {

    arr[k] = L[i];

    items[k] = Litems[i];

    i++;

    k++;

  }

  while (j < n2)

  {

    arr[k] = R[j];

    items[k] = Ritems[j];

    j++;

    k++;

  }

}

void mergeSort(float \*arr, item \*items, int l, int r)

{

  if (l < r)

  {

    int m = (l + r) / 2;

    mergeSort(arr, items, l, m);

    mergeSort(arr, items, m + 1, r);

    merge(arr, items, l, m, r);

  }

}

int main()

{

  int n, i, capacity;

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

  scanf("%d", &n);

  item \*items = (item \*)malloc(n \* sizeof(item));

  printf("Enter the weight and profit of each item:\n");

  for (i = 0; i < n; i++)

  {

    scanf("%d %d", &items[i].weight, &items[i].profit);

  }

  for (i = 0; i < n; i++)

  {

    items[i].x = 0.0;

    items[i].initialIndex = i;

  }

  printf("\nEnter the capacity of the knapsack: ");

  scanf("%d", &capacity);

  float pRatio[n];

  for (i = 0; i < n; i++)

  {

    pRatio[i] = (float)items[i].profit / items[i].weight;

  }

  mergeSort(pRatio, items, 0, n - 1);

  int currentWeight = 0;

  float currentProfit = 0.0;

  for (i = 0; i < n; i++)

  {

    if (currentWeight + items[i].weight <= capacity)

    {

      items[i].x = 1.0;

      currentWeight += items[i].weight;

      currentProfit += items[i].profit;

    }

    else

    {

      items[i].x = (float)(capacity - currentWeight) / items[i].weight;

      currentProfit += items[i].profit \* items[i].x;

      break;

    }

  }

  printf("\nThe pRatio table is: \n");

  for (i = 0; i < n; i++)

  {

    printf("Item %d: %.2f\n", items[i].initialIndex + 1, pRatio[i]);

  }

  printf("\nThe items selected are:\n");

  for (i = 0; i < n; i++)

  {

    if (items[i].x > 0.0)

    {

      printf("Item %d: %.2f\n", items[i].initialIndex + 1, items[i].x);

    }

  }

  printf("The total profit is: %.2f\n", currentProfit);

  return 0;

}