红色部分为修改位置

#include <iostream>

using namespace std;

template<class Typew, class Typep>

class Knap//建立类knap

{

friend Typew Knapsack(Typew\*,Typew\*,Typew,int);//友元函数的声明

private://私有成员声明

Typew Bound(int i);

void Backtrack(int i);

Typew c;//背包容量

int n; //物品数

Typew \*w;//物品重量数组

Typew \*p;//物品价值数组

Typew cw;//当前重量

Typew cp;//当前价值

Typew bestp;//当前最优值

};

template<class Typew, class Typep>

int\* Knap<Typew, Typep>::Backtrack(int i)//友元函数的定义

{

Int \*ans;

Ans=new int[n+1];

if (i>n) {//到达叶结点

bestp = cp;

return;

}

if (cw + w[i] <= c) //搜索左子树

{

cw += w[i];

cp += p[i];

ans[i]=1;

Backtrack(i + 1);

cw -= w[i];

cp -= p[i];

ans[i]=0;

}

if (Bound(i + 1)>bestp)//搜索右子树

Backtrack(i + 1);

}

template<class Typew, class Typep>

Typew Knap<Typew, Typep>::Bound(int i)

{// 计算上界

Typew cleft = c - cw; // 剩余容量

Typep b = cp;

// 以物品单位重量价值递减序装入物品

while (i <= n && w[i] <= cleft) {

cleft -= w[i];

b += p[i];

i++;

}

// 装满背包

if (i <= n)

b += p[i] / w[i] \* cleft;

return b;

}

class Object

{

friend int Knapsack(int \*, int \*, int, int);

public:

int operator<=(Object a)const

{

return (d >= a.d);

}

private:

int ID;//对象好

float d;//收益密度

};

template<class Typew, class Typep>

Typew Knapsack(Typew p[], Typew w[], Typew c, int n)

{

//为Knap::Backtrack初始化

Typew W = 0;

Typew P = 0;

Object \*Q = new Object[n];

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

{

Q[i - 1].ID = i;

Q[i - 1].d = 1.0\*p[i] / w[i];

P += p[i];

W += w[i];

}

if (W <= c)

return P;//装入所有物品

//依物品单位重量排序

Sort(Q，n);

Knap<Typew, Typep> K;

K.p = new Typep[n + 1];

K.w = new Typep[n + 1];

K.x = new Typep[n + 1];

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

{

K.p[i] = p[Q[i - 1].ID];

K.w[i] = w[Q[i - 1].ID];

}

K.cp = 0;

K.cw = 0;

K.c = c;

K.n = n;

K.bestp = 0;

//回溯搜索

K.Backtrack(1);

delete[] Q;

delete[] K.w;

delete[] K.p;

return K.bestp;

}

void main()

{

int \*p;

int \*w;

int c = 0;

int n = 0;

int i = 0;

cout << "请输入背包个数：" << endl;

cin >> n;

p = new int[n + 1];

w = new int[n + 1];

p[0] = 0;

w[0] = 0;

cout << "请输入背包容量：" << endl;

cin >> c;

cout << "请依此输入每个物品的重量：" << endl;

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

cin >> w[i];

cout << "请依此输入每个物品的价值：" << endl;

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

cin >> p[i];

cout << "所能装入背包中总价值最大为" << Knapsack(p, w, c, n) << endl;

}