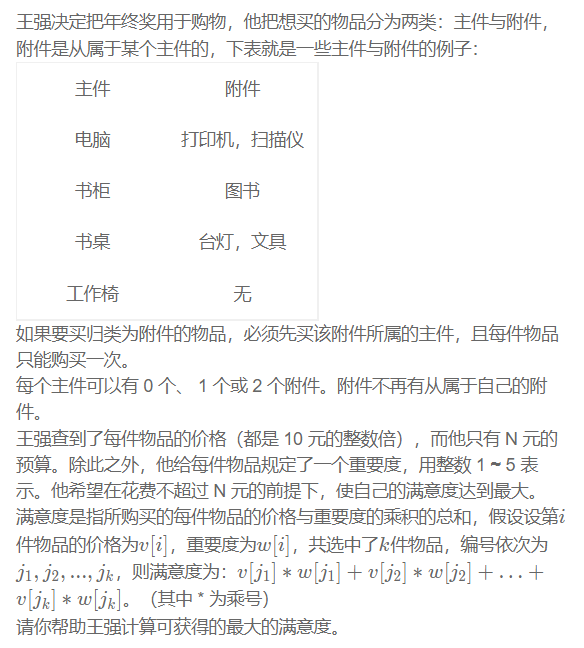
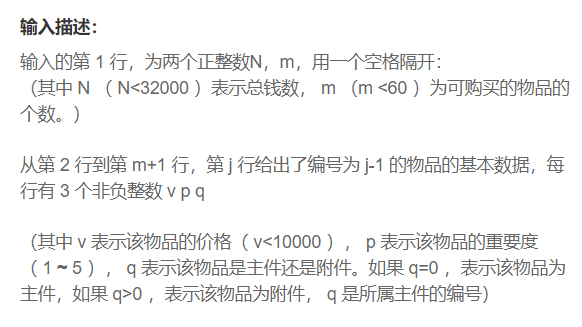
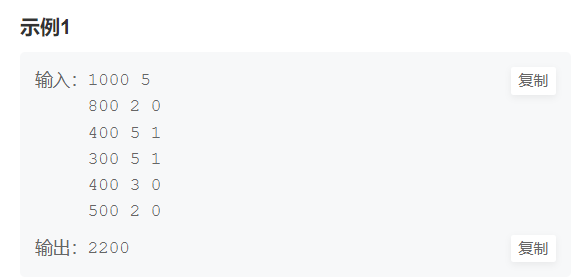
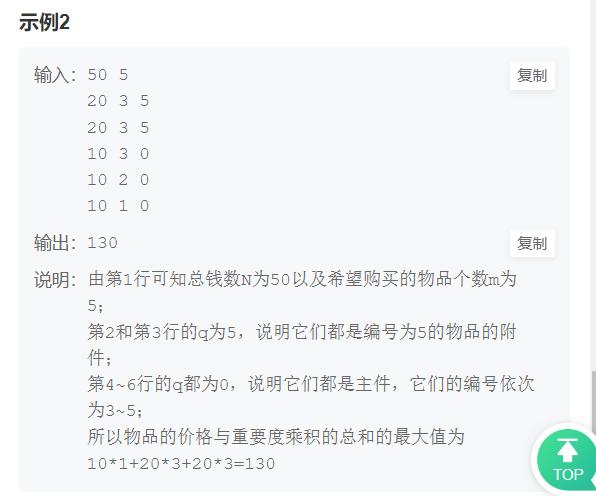
2022.08.30

背包问题 （动态规划）









#include <iostream>

#include <vector>

using namespace std;

int main() {

int N, m;

cin >> N >> m;

N /= 10;

vector<vector<int> > prices(61, vector<int>(3, 0)); // 价格

vector<vector<int> > priceMultiplyPriority(61, vector<int>(3,

0)); // 重要程度

for (int i = 1; i <= m; ++i) {

int a, b, c;

cin >> a >> b >> c;

a /= 10;

b \*= a;

if (c == 0) {

prices[i][0] = a;

priceMultiplyPriority[i][0] = b;

} else {

if (prices[c][1] == 0) {

prices[c][1] = a;

priceMultiplyPriority[c][1] = b;

} else {

prices[c][2] = a;

priceMultiplyPriority[c][2] = b;

}

}

}

// 使用分组背包

vector<vector<int> > dp(m + 1, vector<int>(N + 1, 0));

for (int i = 1; i <= m; ++i) {

for (int j = 1; j <= N; ++j) {

int a = prices[i][0], b = priceMultiplyPriority[i][0];

int c = prices[i][1], d = priceMultiplyPriority[i][1];

int e = prices[i][2], f = priceMultiplyPriority[i][2];

dp[i][j] = j >= a ? max(dp[i - 1][j - a] + b, dp[i - 1][j]) : dp[i - 1][j];

dp[i][j] = j >= (a + c) ? max(dp[i - 1][j - a - c] + b + d,

dp[i][j]) : dp[i][j];

dp[i][j] = j >= (a + e) ? max(dp[i - 1][j - a - e] + b + f,

dp[i][j]) : dp[i][j];

dp[i][j] = j >= (a + c + e) ? max(dp[i - 1][j - a - c - e] + b + d + f,

dp[i][j]) : dp[i][j];

}

}

cout << dp[m][N] \* 10 << endl;

}