Динаміка

Динамічне програмування . Купування квитків

<https://www.e-olymp.com/ru/problems/799>

|  |  |
| --- | --- |
| Ім'я файлу програми: | **Bilet.\*** |
| Ім'я вхідного файлу: | **Bilet.dat** |
| Ім'я вихідного файлу: | **Bilet.dat** |
| Максимальний час роботи на одному тесті: | **5 секунд** |
| Максимальна оцінка за завдання: | **100 балів** |

За квитками на прем'єру нового мюзиклу вишикувалася черга з *N* людей, кожний з яких хоче купити 1 квиток. На всю чергу працювала тільки одна каса, тому продаж квитків йшов дуже повільно, приводячи людей черги у відчай. Найкмітливіші швидко відмітили, що, як правило, декілька квитків в одні руки касир продає швидше, ніж коли ці ж квитки продаються поодинці. Тому вони запропонували декільком людям, які стоять підряд віддавати гроші першому з них, щоб він купив квитки на всіх.

Проте для боротьби із спекулянтами касир продавала не більше 3-х квитків в одні руки, тому домовитися таким чином між собою могли лише 2 або 3 підряд вартих людини.

Відомо, що на продаж *i-ій* людині з черги одного квитка касир витрачає Ai секунд, на продаж двох квитків — *Bi* секунд, трьох квитків — *Ci* секунд. Напишіть програму, яка підрахує мінімальний час, за який могли бути продані квитки для всіх людей черги.

Зверніть увагу, що квитки на групу людей, що об'єдналися, завжди купує перший з них. Також ніхто в цілях прискорення не купує зайвих квитків (тобто квитків, які нікому не потрібні).

Формат вхідних даних

У вхідному файлі записано спочатку число N — кількість покупців в черзі (1<=N<=5000). Далі йде N трійок натуральних чисел Ai*,* Bi*,* Ci*.* Кожне з цих чисел не перевищує 3600. Люди в черзі нумеруються починаючи від каси.

Формат вихідних даних

У вихідний файл виведіть одне число — мінімальний час в секундах, за яке могли бути обслужені всі покупці.

Приклади

|  |  |
| --- | --- |
| Bilet.dat | Bilet.sol |
| 55 10 152 10 155 5 520 20 120 1 1 | 12 |
| 23 4 51 1 1 | 4 |

*N*=5

|  |  |  |  |
| --- | --- | --- | --- |
| ***i*** | ***Ai*** | ***Bi*** | ***Ci*** |
| 1 | 5 | 10 | 15 |
| 2 | 2 | 10 | 15 |
| 3 | 5 | 5 | 5 |
| 4 | 20 | 20 | 1 |
| 5 | 20 | 1 | 1 |

D[i]= min ( D[i-1]+*Ai*, D[i-2]+ *Bi-1*, D[i-3]+ *Ci-2* )

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| D[1] | D[2] | D[3] | D[4] | D[5] |
| 5 | 7 | 5 | 6 | **12 – відповідь завдання** |

d[0]= 0;

d[1]= а[1];

d[2]= Мінімальне(а[1]+a[2], b[1]);

Для i від 3 до n пц

d[i]= Мінімальне(d[i-1]+ а[i],Мінімальне(d[i-2]+ b[i-1], d[i-3]+ с[i-2]));

d[0]= 0;

d[1]= а[1];

d[2]= min(а[1]+a[2], b[1]);

for(int i=3;i<=n;i++)

d[i]= min(d[i-1]+ а[i],min(d[i-2]+ b[i-1], d[i-3]+ с[i-2]));

#include<fstream>

using namespace std;

ifstream cin("input.txt");

ofstream cout("output.txt");

int main()

{

int n,i,a[5000],b[5000],c[5000],d[5000];

in>>n;

for(i=1;i<=n;i++) cin>>a[i]>>b[i]>>c[i];

d[0]= 0; d[1]= a[1]; d[2]= min(a[1]+a[2],b[1]);

for(i=3;i<=n;i++) d[i]=min(d[i-1]+a[i],min(d[i-2]+b[i-1],d[i-3]+c[i-2]));

cout<<d[n]<<endl;

}

Найкоротші шляхи

Нехай є n міст, пронумерованих числами від 1 до n. Для кожної пари міст із номерами і, j у таблиці a[і][j] зберігається ціле число - ціна прямого авіаквитка з міста i у місто j. Вважається, що рейси існують між будь-якими містами, a[і,і] = 0 при всіх і, a[і][j] може відрізнятися від a[j,і]. Найменшою вартістю проїзду з і в j вважається мінімально можлива сума цін квитків для маршрутів (у тому числі з пересадженнями), що ведуть з і в j. (Вона не перевершує a[і][j], але може бути менше).

У пропонованих нижче задачах потрібно знайти найменшу вартість проїзду для деяких пар міст при тих чи інших обмеженнях на масив a і на час роботи алгоритму.

Припустимо, що не існує замкнутих маршрутів, для яких сума цін негативна. Передбачається, що ця умова (відсутність циклів з негативною сумою) виконана.

1) Знайти найменшу вартість проїзду з 1-го міста в усі інші .

Позначимо через Мінвар(1,s,к) найменшу вартість проїзду з 1 у s менш чим з k пересадженнями. Тоді виконується таке співвідношення:

Мінвар (1,s,k+1) = найменшому з чисел Мінвар(1,s,k) і

Мінвар(1,і,k) + a[і][s] (і=1..n)

Як відзначалося вище, шуканою відповіддю є Мінвар (1,і,n) для всіх і=1..n.

((0,20,3,4,5,6),

 (20,0,5,6,7,8),

 (3,5,0,6,7,8),

 (4,6,6,0,8,8),

 (5,7,7,8,0,9),

 (6,8,8,8,9,0));

1 2 -----20

1-3 ---- 3 , 3-2---- 5= 8

min(20,8)=8

флойд

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

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

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

a[i][j]=min(a[i][j].a[i][k]+a[k][j];

Set

#include <iostream>

#include <set> // заголовочный файл множеств и мультимножеств

#include <iterator>

using namespace std;

int main()

{

 set<char> mySet; // объявили пустое множество

 // добавляем элементы в множество

 mySet.insert('I');

 mySet.insert('n');

 mySet.insert('f');

 mySet.insert('i');

 mySet.insert('n');

 mySet.insert('i');

 mySet.insert('t');

 mySet.insert('y');

 copy( mySet.begin(), mySet.end(), ostream\_iterator<char>(cout, " "));

cout<<endl;

 if (mySet.count('n') != 0) {

 cout << "n is in set}";

 }

 else {

 cout << "n is not in set";

 }

 mySet.erase('n');

cout<<endl;

 if (mySet.count('n') != 0) {

 cout << "n is in set}";

 }

 else {

 cout << "n is not in set";

 }

 return 0;

}