Упражнения: Динамично оптимиране

Задача 1. Алея

Алея в градската градина на град X има дължина L и трябва да бъде павирана с правоъгълни плочи. Всяка плоча има ширина, колкото алеята, но дължините на плочите са различни. Плочите са N вида и видовете са номерирани от 1 до N. Разполагаме с достатъчно плочи от всеки вид. Определете по колко различни начина може да се павира алеята, като се използват съществуващите видове плочи.

На първия ред на стандартния вход се въвеждат две положителни цели числа, разделени с интервал - L, дължината на алеята и N - броят на видовете плочи. На втория ред се въвеждат N на брой положителни цели числа, разделени с интервал, d_1 d_2 ... d_n, където d_i е равно на дължината на плочата от вид i, i = 1, ..., N.

Програмата да извежда едно положително цяло число, равно на броя начини, по които алеята може да павира.

Пример:

Вход	Изход
5 3 2 1 3	13

Ограничения:

- 0<L<=700
- 0<N<=250
- 0<di<1000, за всяко i=1...N.
- di ≠ dj, за всяко i=1...N.

Резултатът може да има най-много 300 цифри.

Подсказки:

Пребройте различните подреждания на плочите според това каква плоча завършва на L-тия метър, ако позволява дължината ѝ. Ако плочата с пореден номер i, завършва на L-тия метър, то броя на различните начини за това е броят на различните подреждания на плочите до L - дължината на i-тата плоча. И така общия брой е сумата от различните подреждания за всеки вид плоча. Използвайте един масив, в който да пазите дължините на различните плочи – p[], и един масив, в който да пазите броя на подрежданията – b[].

Задача 2. Триъгълник от числа

Нека е даден следния триъгълник от числа:

				7				
			3		8			
		8		1		0		
	2		7		4		4	
4		5		2		6		5

Напишете програма, която пресмята най-голямата възможна сума от числа, разположени на някои от пътищата, започващи от най-горната точка на триъгълника и завършващи в точка от основата на триъгълника. Изисква се пътищата да са такива, че при всяка от стъпките движението да се осъществява надолу — в посока по диагонала наляво или по диагонала надясно.

Пример:

Вход	Изход
0 7 0 0 0 0 0 3 8 0 0 0 0 8 1 0 0 0 0 2 7 4 4 0 0 4 5 2 6 5	30

Подсказки:

За да решим задачата в общия случай, нека да запишем данните в двумерен масив D[i][j] с размери Nx(N + 1), i = 0,1,...,N-1, j = 0,1,... ,N. По технически причини, за по-лесно боравене с индексите в програмата, този масив го дефинираме с един стълб в повече, като допълнителният нулев стълб остава зареден с нули. При конкретно зададените числа масивът представя следната таблица от 5 реда и 6 стълба (N = 5):

```
0 7 0 0 0 0
0 3 8 0 0 0
0 8 1 0 0 0
0 2 7 4 4 0
0 4 5 2 6 5
```

За всеки елемент D[i] [j], включен в дадената триъгълна кон-фигурация от числа, да пресметнем най-голямата стойност R[i][j] , която може да се постигне, като се движим според правилата, тръг-вайки от върха. Стойностите пресмятаме последователно по редове. Очевидно имаме:

```
R[0][1] = D[0][1];
R[i][j] = max{D[i][j] + R[i - 1][j - 1], D[i][j] + R[i - 1][j]}
```

за всички i = 1,..., N-1 и j = 1,..., i+1. Остава да намерим най-голямата стойност в последния ред на масива R.

Спомагателният масив P[i][j] служи за възстановяване на самия път, по който се достига най-голямата сума от търсения вид.

Задача 3. Управление на врата

В един ресторант се срещат N бандити. Бандитът с номер i, i = 1,2,..., N идва в момента от време Ti и носи Pi долара. Входната врата на ресторанта има K състояния, разли-чаващи се по степента на отвореност. Състоянието на вратата може да се променя с една единица за всяка една единица време, т.е. степента на отваряне на вратата или се увеличава с едини-ци, или се намалява с единица, или остава в същото състояние. В началния момент от време вратата е затворена (състояние 0). Бандитът с номер i може да влезе в ресторанта само ако врата-та с отворена специално за него, т. е. когато степента на отвореност на вратата съвпада със степента S_i на пълнота на бандита. При едновременно идване на няколко бандити с еднаква пълнота, съвпадаща със степента на отвореност на вратата, влизат всичките бандити с тази пълнота. Ако в момента на идване на един бандит състоянието на вратата не съвпада със степента на пълнота на бандита, той си отива и повече не се връща. Ресторантът работи в течение на време T. Напишете прог-рама, която да покаже по какъв начин вратата да се отваря или затваря във всяка стъпка от времето, така че в ресторанта да се съберат бандити с максимално количество долари.

Входни данни за задачата са:

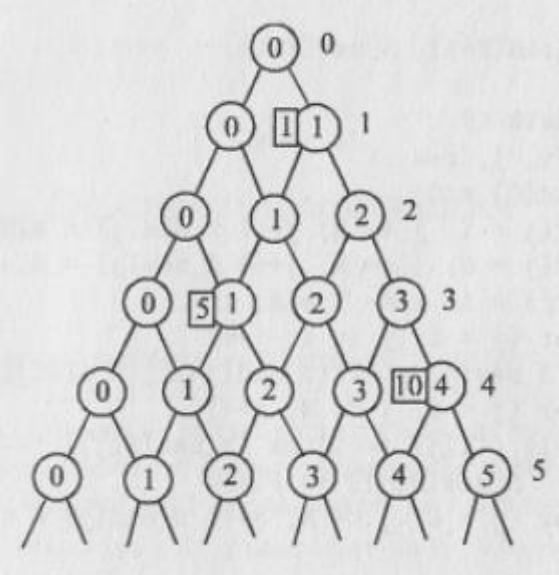
- целите числа N, K и T (примерни ограничения: 1 <= N <= 100, 1 < =K <= 100, 0 < =T<= 3000);
- моментите от време на пристигане на бандитите T_1, T_2, ... , T_N, зададени като цели числа от интервала [1,T].
- количеството долари, които носи всеки бандит: P_1, P_2, ... ,P_N, зададени като цели числа (например между 0 и 300).
- степента на пълнота на всеки от бандитите: S_1, S_2, ... ,S_N, които са цели числа такива, че $1 < S_i < K$ за всяко i =1,2,..., N.

Пример

Вход	Изход
N=3 K=5 T=5 T_1=3 T_2=4 T_3=1 P_1=5 P_2=10 P_3=1 S_1=1 S_2=4 S_3=1	11

Подсказки:

Състоянието на отвореност на вратата изобразяваме като триъгълна решетка. На изображението по-долу са представени входните данни за задачата. Всеки връх определя степента q на отвореност в момента време t. Моментите от време са отбелязани като числа отдясно на решетката. При някои от върховете в правоъгълна рамка е дадено количеството долари които има този бандит (с пълнота q), който идва в момента време t. За всеки връх това количество долари ще наричаме тегло на върха. За върховете, които нямат съпоставено количество долари, приемаме, че теглото им е нула. За да решим задачата, трябва да намерим път по решетката, започващ от най-горния връх и минаващ през върхове така, че сумата от теглата на върховете по пътя да е максимална. За дадения при-мер, тази максимална стойност лесно може да бъде намерена и тя е равна на 11.



За програмната реализация ще отбележим, че не е необходимо да се пазят оценките за всеки връх. За да ги намерим в момента t, е необходимо да ползваме само стойностите им в предишния момент t-1. Това се осъществява в програмата чрез двата едномерни масива S_old и S_new. В основния цикъл променливата i пробягва моментите от време. В тялото на този цикъл първоначално елементите на масива S_new се зареждат с по-голямата от двете „горни“ стойности. След това програмата преглежда времената на пристигане на бандитите. Ако намери бандит, който пристига в текущия момент (T[j]==i) и установи, че съществува степен на отвореност на вратата, съответстваща на пълнотата на бадита (S_new[S[j]] != MINVALUE), програмата коригира стойността нп S_new.