

La maleta de medicamentos

Tenemos hasta 5 tipos de medicamentos y queremos llenar una maleta de cajas de estos medicamentos, maximizando el valor total de los medicamentos incluidos.

La caja de cada tipo de medicamento tiene un volumen determinado en centímetros cúbicos. Cada tipo de medicamento tiene un valor. Y para cada tipo de medicamento necesitamos un número mínimo de cajas.

Debes escribir un programa que calcule el número de cajas de cada tipo de medicamento que debemos colocar en la maleta, de forma que se consiga **el mayor valor posible del total de medicamentos**, respetando el número mínimo de cajas de cada tipo y haciendo que la suma de los volúmenes de todas las cajas no sobrepase el volumen total de la maleta (suponemos que las cajas de medicamentos son deformables y se pueden acomodar en los huecos restantes en la maleta, de forma que al final la maleta queda completamente llena).

Deberás imprimir el valor total de la mejor combinación de número de cajas de medicamentos.

Ejemplo

Supongamos que tenemos una maleta con una capacidad de 10.000 cm^3 .

Y tenemos 3 tipos de medicamentos con las siguientes características:

Medicamento 1	Medicamento 2	Medicamento 3
Volumen: 135 cm^3 Valor: 8 € Mínimo de 10 cajas	Volumen: 190 cm^3 Valor: 11 € Mínimo de 4 cajas	Volumen: 20 cm^3 Valor: 1 € Mínimo de 9 cajas

Con estas restricciones, la mejor combinación de cantidades de medicamentos a añadir en la maleta es:

- 64 cajas del medicamento 1
- 6 cajas del medicamento 2
- 11 cajas del medicamento 3

El volumen total de las cajas es: $64 \cdot 135 + 6 \cdot 190 + 11 \cdot 20 = 10.000 \text{ cm}^3$

El valor total de los medicamentos es $64 \cdot 8 + 6 \cdot 11 + 11 \cdot 1 = 589 \text{ €}$

Formato de la entrada

La primera línea indicará el número de medicamentos (n) y la capacidad de la maleta.

Y después tendremos tantas líneas como número de medicamentos. Cada línea tendrá tres números enteros: el volumen de la caja, el valor en euros y el número mínimo de cajas.

El ejemplo anterior tendría la siguiente entrada:

```
3 10000
135 8 10
190 11 4
20 1 9
```

Formato de la salida

Se debe imprimir un número entero indicando el valor total de la mejor combinación de medicamentos.

El ejemplo anterior tendría la siguiente salida:

```
589
```

Puntuación

- **Test 1 (10 puntos):** Los datos de prueba tendrán solo 1 tipo de medicamento ($n = 1$ para todos los ejemplos de prueba).
- **Test 2 (20 puntos):** Los datos de prueba tendrán 2 tipos de medicamentos ($n = 2$ para todos los ejemplos de prueba).
- **Test 3 (70 puntos):** Los datos de prueba tendrán un número de medicamentos variado, como máximo de 5 medicamentos ($n \leq 5$).

Monumentos

En la ciudad que estamos visitando vemos muchos monumentos, y en todos ellos aparecen inscripciones con el año del monumento escrito en números romanos. Queremos saber cuáles son los monumentos más antiguos de los que hemos visto.

Como sabes, en los números romanos se suman los valores de las diferentes letras, siendo M=1000, D=500, C=100, L=50, X=10, V=5, I=1, pero teniendo en cuenta que hay unas excepciones CM=900, CD=400, XC=90, XL=40, IX=9, IV=4.

Escribe un programa que tome como entrada un número de versión y una serie de números romanos que son los años de los monumentos que vamos viendo y que acaba con el texto RIP (siempre después de haber introducido al menos dos números), y proporcione:

- para la versión 0 debe dar como resultado el número de monumentos vistos;
- para la versión 1 debe dar como resultado en la salida el año, en números “modernos” (en base 10), del monumento más antiguo, suponiendo que los números romanos introducidos no contienen ninguna de las combinaciones-excepciones CM, CD, XC, XL, IX, IV (se asume que todos los números romanos son correctos);
- para la versión 2 debe dar como resultado en la salida, en una sola línea, los años, en números romanos, de los dos monumentos más antiguos, asumiendo que se introduce una secuencia de números romanos cualesquiera correctos.

En cualquier caso la salida debe terminar en un salto de línea.

Puntuación:

- + 10 puntos por dar el resultado correcto de la versión 0.
- + 30 puntos por dar el resultado correcto de la versión 1.
- + 60 puntos por dar el resultado correcto de la versión 2.

Ejemplos:

Entrada: `0 MM MCLXVII MDCCXV RIP`

da como resultado: `3`

Entrada: `1 MM MCLXVII MDCCXV RIP`

son los años 2000, 1167, 1715 y da como resultado: `1167`

Entrada: `2 MCDXCII MMXXII CMXLIX MCMLXXXIV RIP`

son los años 1492, 2022, 949, 1984 y da como resultado: `CMXLIX MCDXCII`

OOIICCVV

Queremos hacer un programa que muestre nuestra alegría de que llega la OICV y permita imprimir esas siglas con distintos números de letras.

Tendremos cuatro números enteros n_o , n_i , n_c y n_v que indicarán el número de "O"s, el número de "I"s, el número de "C"s y el número de "V"s.

Por ejemplo, si los números son:

$$n_o = 2$$

$$n_i = 1$$

$$n_c = 3$$

$$n_v = 2$$

deberemos escribir la palabra OOICCCVV que tiene 2 "O"s, 1 "I", 3 "C"s y 2 "V"s.

Leeremos esos cuatro números y tendremos que escribir la palabra resultante.

Entrada

La entrada comienza con un número t que indica el número de casos a procesar. Cada una de las siguientes líneas contiene los enteros n_o , n_i , n_c y n_v de aquel caso.

$$0 \leq t \leq 1000$$

$$0 \leq n_o, n_i, n_c, n_v \leq 100$$

Salida

Para cada caso escribid una línea con la palabra codificada por los cuatro enteros.

Puntuación

- **Test 1 (10 puntos):** $n_o = n_i = n_c = n_v = 1$
- **Test 2 (20 puntos):** $1 \leq n_o = n_i = n_c = n_v \leq 10$
- **Test 3 (50 puntos):** $1 \leq n_o, n_i, n_c, n_v \leq 100$
- **Test 4 (20 puntos):** $0 \leq n_o, n_i, n_c, n_v \leq 100$ (algunos o todos los enteros pueden ser 0)

Ejemplos

Entrada:

```
4
1 1 1 1
2 2 2 2
1 3 4 2
0 2 0 1
```

Salida:

```
OICV
OOIICCW
OIIICCCCW
IIV
```

¡Están vivos!

Daniel Nudos, un importante investigador de la Universidad, ha realizado un extraño descubrimiento durante sus investigaciones sobre nuevos algoritmos. En un vector de enteros relleno de ceros y abandonado durante varios días ha empezado a crecer una nueva fauna que parece alimentarse de dichos ceros. Analizándolos en detalle ha podido descubrir que pasan por varias fases durante su vida: la niñez (fase '1'), la edad adulta (fase '2') y adulto con hijo (fase '3'). También ha podido descubrir que, en cada iteración de la vida, dependiendo de las fases de sus vecinos, pueden evolucionar, siempre respetando el orden de los índices del vector, es decir, de izquierda a derecha. Las reglas que siguen son las siguientes :

1. Si un niño (1) tiene alimento (0) a su lado (a su izquierda o su derecha), pasa a estado adulto (2). En caso contrario, muere (0).
2. Si dos adultos adyacentes (22) tienen alimento (0) en su casilla izquierda, en dicha casilla nace un niño (1) y ellos se convierten en adultos con hijos (33).
3. Si dos adultos adyacentes (22) no tienen alimento en su casilla izquierda pero sí tienen en su casilla derecha, en dicha casilla nace un niño (1) y ellos se convierten en adultos con hijos (33).
4. Si un adulto con hijos (3) tiene comida (0) a una distancia de 1 o 2 casillas, pasa a adulto (2), pero si tiene la comida más lejos de 2 casillas, entonces muere (0).

Escribe un programa que simule dicha vida y permita saber cómo evolucionará un estado inicial determinado en un plazo de N iteraciones.

Se debe tener en cuenta que fuera de los límites del vector no hay ni alimento ni individuos.

Entrada

La entrada constará de 2 líneas. En la primera línea estará el tamaño del vector (N) y el número de iteraciones a realizar (It) y en la segunda línea, N números representando el estado inicial del vector.

$1 \leq N \leq 100$

$1 \leq It \leq 100$

Salida

It+1 líneas de N números representando los estados del vector en cada iteración de la vida, incluyendo el estado inicial.

Puntuación

Test 1 (10 puntos): $N \geq 4$. $It = 1$. Solo se aplica la **regla 1**. A lo largo de toda la simulación, siempre quedan al menos 2 ceros al principio del vector y 2 ceros al final del vector.

Test 2 (30 puntos): $N \geq 4$. A lo largo de toda la simulación, siempre quedan al menos 2 ceros al principio del vector y 2 ceros al final del vector.

Test 3 (60 puntos): Sin restricciones.

Ejemplos

Entrada

```
6 1
0 0 1 2 0 0
```

Salida

```
0 0 1 2 0 0
0 0 2 2 0 0
```

Entrada

```
10 6
0 0 0 0 0 2 2 0 0 0
```

Salida

```
0 0 0 0 0 2 2 0 0 0
0 0 0 0 1 3 3 0 0 0
0 0 0 0 2 2 2 0 0 0
0 0 0 1 3 3 2 0 0 0
0 0 0 2 2 2 2 0 0 0
0 0 1 3 3 3 3 1 0 0
0 0 2 2 0 0 2 2 0 0
```

Entrada

```
12 5
2 2 0 0 1 3 3 1 2 2 2 0
```

Salida

2	2	0	0	1	3	3	1	2	2	2	0
3	3	1	0	2	2	0	0	2	3	3	1
0	2	2	1	3	3	0	0	2	2	0	0
1	3	3	0	2	2	0	1	3	3	0	0
0	2	2	1	3	3	0	2	2	2	0	0
1	3	3	0	2	2	1	3	3	2	0	0

Zona de bajas emisiones

La Comunidad Europea ha puesto en marcha un proceso de control de emisiones contaminantes en todo su territorio. En las ciudades más importantes existe una *Zona de Bajas Emisiones* (ZBE), donde la circulación de vehículos está sujeta a ciertas restricciones.

Existen cinco categorías de vehículos, que deben circular con un distintivo (etiqueta) que indica bajo qué condiciones puede circular dicho vehículo por la ZBE. De menos a más emisiones, estas categorías se ordenan así:

CERO - ECO - C - B - SIN

(SIN son vehículos sin etiqueta)

Se ha establecido el siguiente protocolo de actuación:

Si el nivel de contaminación detectado en una determinada franja horaria supera un determinado umbral en al menos dos estaciones medidoras diferentes, se considera restringida la circulación de ciertos tipos de vehículos dentro de la ZBE mientras se mantenga esa situación.

Se puede producir uno de los siguientes escenarios:

Tabla 1

Escenario	Umbral
1	nivel < 180 $\mu\text{g}/\text{m}^3$
2	nivel \geq 180 $\mu\text{g}/\text{m}^3$
3	nivel \geq 200 $\mu\text{g}/\text{m}^3$
4	nivel \geq 400 $\mu\text{g}/\text{m}^3$
5	nivel \geq 500 $\mu\text{g}/\text{m}^3$

Por ejemplo, si hay 3 estaciones, con medidas 573, 389 y 421, se da el escenario 4, es decir, el peor de los escenarios para el cual al menos dos estaciones superan su umbral mínimo. Si no hay dos o más medidas por encima de 180, estaríamos en el escenario 1.

Para ello, las ciudades disponen de una red de estaciones detectoras de polución que monitorizan el aire dentro de la ZBE. Estas estaciones indican, para cada franja horaria de un día determinado, el nivel de contaminantes en el aire (en $\mu\text{g}/\text{m}^3$). Por ejemplo:

Tabla 2

Día	Estación	0h	2h	4h	6h	8h	10h	12h	14h	16h	18h	20h	22h
1	11	133	37	25	36	140	250	191	89	81	87	273	503
1	7	164	228	234	38	136	447	153	52	46	65	189	296
1	4	386	357	327	52	155	450	246	43	48	72	328	594

La primera columna indica el día del mes, la segunda la estación medidora y las columnas restantes el nivel de contaminantes detectados para cada franja horaria (se indica la hora de comienzo de la franja). Con estos datos, tendríamos los siguientes escenarios:

Escenario	1	3	3	1	1	4	2	1	1	1	3	5
-----------	---	---	---	---	---	---	---	---	---	---	---	---

Todos los conductores pueden obtener información sobre el nivel de contaminación en tiempo real gracias a una aplicación para móviles gratuita. Además, la ciudad dispone de un sistema de cámaras que registran constantemente la entrada y salida de vehículos de la ZBE. Se ha puesto en marcha un sistema de recompensas y penalizaciones (que se traduce en bajadas/subidas del impuesto de circulación), en base a una puntuación. Cada vehículo que se encuentre circulando por la ZBE recibirá una puntuación positiva (recompensa) o negativa (penalización) por cada hora que se encuentre dentro de la ZBE según esta tabla:

Tabla 3

Etiqueta	Escenario 1	Escenario 2	Escenario 3	Escenario 4	Escenario 5
CERO	0	0	0	0	1
ECO	0	0	0	1	-1
C	0	0	1	-1	-2
B	0	1	-1	-2	-3
SIN	1	-1	-2	-3	-4

Las penalizaciones sólo se aplican a partir de la segunda hora que el vehículo permanece en la ZBE en un escenario donde se le penaliza. Esto da tiempo al conductor para abandonar la ZBE cuando la cosa 'se pone fea' sin recibir penalización inmediata.

Caso 1. (35 puntos) Dada una tabla de emisiones diarias y el protocolo de actuación indicado, los ayuntamientos quieren saber:

Cuál es el escenario para cada franja horaria de un día determinado (p. ej., escenarios para el día 1 de la tabla 2, arriba).

ENTRADA (Caso 1)

3	3	5				
1	1	505	699	34	88	614
1	2	535	391	493	84	472
1	3	373	401	34	480	285
3	1	55	468	98	611	367
3	2	313	367	187	688	673
3	3	328	393	456	510	30
2	1	678	386	214	425	227
2	2	77	387	582	348	25
2	3	505	587	260	573	207
1						
3						
2						

- Primera línea: N M H

1 <= N <= 31 : Número de días en la tabla de registros

2 <= M <= 24 : Número de estaciones detectoras

1 <= H <= 24 : Número de franjas horarias

(los días y estaciones medidoras pueden no aparecer ordenados)

- Sigüientes NxM líneas: registros diarios de cada estación:

<dia n> <estación m> <nivel franja 1> ... <nivel franja H>

- Sigüiente línea: 1 (identifica el caso a resolver)

- Sigüientes líneas hasta fin de la entrada: Día para el cual se desea saber cuales son los escenarios.

SALIDA: tantas líneas como días se solicitan en la entrada, con este formato:

<dia> <escenario franja 1> <escenario franja 2> ... <escen. franja H>

Para el ejemplo de entrada anterior:

```
3 3 3 2 5 3
2 5 3 3 4 3
```

Caso 2. (65 puntos) Los ayuntamientos necesitan calcular, dados los rangos horarios que un vehículo de una categoría permanece en la ZBE, cuál será su puntuación. Por ejemplo, un vehículo con etiqueta B, que el día 1 entra a la ZBE a las 9h y salga a las 13h, y vuelva a entrar a las 20h y salga a las 23h,

según los datos de la tabla 2 (arriba), la puntuación del vehículo será:

```
[9-13] : -1    [9-10]: 0, [10-11]: (-2), [11-12]: -2, [12-13]: +1
[20-23] : -4   [20-21]: (-1), [21-22]: -1, [22-23]: -3
Total:      -5
```

(en rojo, las penalizaciones que no se aplican por ser la primera hora con penalización)

ENTRADA (Caso 2)

El número de franjas horarias será fijo $H=12$, es decir, cada franja será de dos horas, como en la Tabla 2:

```
2 3 12
1 1 262 473 398 89 446 71 229 201 166 445 176 154
1 2 90 389 458 485 276 431 438 174 522 561 109 381
1 3 305 421 392 260 496 325 313 486 28 221 110 593
2 1 463 339 372 104 600 494 81 154 127 532 106 110
2 2 357 191 256 503 555 152 214 58 307 41 476 469
2 3 236 519 237 56 568 473 34 157 518 465 553 535
2
B 2 1 9 12 2 20 22
C 3 1 10 11 2 0 5 2 7 12
ECO 3 1 10 11 2 0 5 2 7 12
CERO 3 1 10 11 2 0 5 2 7 12
SIN 3 1 10 11 2 0 5 2 7 12
```

- Las $N \times M + 1$ primeras líneas son como en el caso 1
- 2 : identifica el caso 2
- Sigüientes líneas hasta fin de la entrada: registros de entrada/salida de vehículos. Cada registro de entrada/salida consta de tres valores: día, hora de entrada y hora de salida:

```
<tipo> <nº registros> <día> <entrada> <salida> ... <día> <entrada>
<salida>
```

- Las horas de entrada y salida están entre 0 y 23. En un rango horario, la hora de entrada siempre será menor que la hora de salida. Debes considerar cada rango horario como independiente de los demás, sin considerar que estén ordenados, que se solapen, que haya rangos consecutivos, etc.

SALIDA: puntuación para cada vehículo, una por línea. Para la entrada anterior:

-4
2
1
2
-17