

## SOPA DE LETRAS

### DESCRIPCION

De camino a la OMI habrás pasado horas interminables en el autobús y para pasar el rato te has puesto a resolver una sopa de letras. Como ya te imaginarás, una sopa de letras consiste de un arreglo rectangular en cada una de cuyas posiciones se encuentra una letra minúscula. Además, se tiene una lista de palabras (formadas cada una exclusivamente por letras minúsculas) que se deben de buscar en el arreglo. Una palabra puede aparecer en línea recta en el arreglo en ocho posibles direcciones comenzando desde su primer letra: hacia arriba, hacia abajo, hacia la derecha, hacia la izquierda y en las cuatro direcciones diagonales. Además, una palabra puede aparecer en más de un lugar.

### PROBLEMA

Deberás escribir un programa que reciba como entrada las dimensiones del arreglo rectangular, las letras que aparecen en el mismo, el tamaño de la lista y la lista de palabras a buscar y que determine cuántas veces aparece cada una de estas palabras en el arreglo.

### ENTRADA

Tu programa deberá leer del teclado de la PC los siguientes datos

- En la primera línea los números  $0 < N, M \leq 100$  los cuales representan el número de renglones y el número de columnas del arreglo, respectivamente.
- En cada una de las siguientes  $N$  líneas una cadena de  $M$  letras minúsculas, las cuales representan el arreglo.
- En las siguientes líneas el número  $0 < K \leq 20000$  que representa el tamaño de la lista de palabras.
- En cada una de las siguientes  $K$  líneas una palabra  $P_i$  de 2 a 100 letras minúsculas.

### SALIDA

Tu programa deberá escribir en la pantalla de la computadora  $K$  líneas y en cada una de ellas debe aparecer la cantidad  $X_i$  de veces que aparece la palabra  $P_i$

ENTRADA	SALIDA
2 3	6
aal	2
ala	
2	
al	
ala	

La palabra *a/* aparece 6 veces en la sopa. Las 6 veces son:

aal	aal	aal	aal	aal	aal
ala	ala	ala	ala	ala	ala

La palabra *ala* aparece 2 veces, ambas están en el último renglón, la primera se puede leer comenzando de izquierda a derecha y la segunda aparición se lee de derecha a izquierda.

## RULETA

### DESCRIPCIÓN

El juego de la ruleta ha sido objeto de innumerables intentos de descubrir alguna flaqueza que permitiera diseñar un método para obtener ganancias sistemáticas. Numerosos autores han escrito tratados de cómo quebrar la banca. Norman Leigh escribió una obra clásica donde expone como, junto con doce compañeros, quebraron la banca de Montecarlo. La obra, magistralmente escrita, cuenta hechos "verídicos". Pero con el auxilio de la ciencia de la computación podemos ver que esos hechos tal como los relata Leigh no pueden haber sucedido.

El método que Leigh relata en su libro es el Labouchere inverso. El apostador comienza anotando en una libreta los números 1, 2, 3 y 4. Siempre apuesta una cantidad igual a la suma de los números de los extremos de la lista. Si gana agrega al final de la lista un número de igual valor al de su última apuesta. Si pierde tacha los dos números de los extremos de la lista. Si en cualquier momento la lista se vacía, el jugador reinicia el método anotando los números 1, 2, 3 y 4 en su libreta. Si en cualquier momento la lista contiene sólo un número, la apuesta que se realiza es igual a ese número. Si la apuesta llega a superar a la máxima permitida, el jugador también reinicia el método anotando una nueva lista de 1, 2, 3 y 4.

Recordamos que la ruleta europea consta de los números del 0 al 36 y que, jugando a par se gana si se obtiene el número par distinto de cero, el que juega a impar gana si el número es impar y en caso de salir cero ambos pierden. La apuesta máxima permitida es de \$100, recuerde que cuando se supere esta apuesta, el método se reinicia. En cada apuesta se gana o se pierde el monto apostado.

Con dos jugadores, uno jugando a par y otro a impar el método se comporta así en unas bolas de ejemplo (el primer renglón muestra las bolas de ejemplo, al último la ganancia acumulada y los demás la lista en cada momento):

<i>Par</i>	31	22	26	11	4	36	0	4	<i>Impar</i>	31	22	26	11	4	36	0	4
1	2	2	2	3	3	3	5	5	1	1	2	3	3	1	2	1	2
2	3	3	3	5	5	5	8	8	2	2	3		3	2	3	2	3
3		5	5		8	8		13	3	3	4			3		3	
4			7			11			4	4				4		4	
										5							
<i>Ganancia acumulada</i>	-5	0	+7	-2	+6	+17	+3	P=+16	<i>Ganancia acumulada</i>	+5	-1	-7	-4	-10	-15	-20	Q=-25

### PROBLEMA

Debes escribir un programa que calcule la ganancia o pérdida lograda luego de jugarse cierta cantidad de bolas de ruleta

### ENTRADA

Tu programa deberá leer del teclado de la PC los siguientes datos:

- En la primera línea el número  $0 < N \leq 30,000$  de tiradas de ruleta
- En cada una de las siguientes  $N$  líneas un número entero entre 0 y 36 representando la bola obtenida en cada uno de los  $N$  giros de la ruleta.

### SALIDA

Tu programa deberá escribir en la pantalla de la PC dos enteros  $P$  y  $Q$  separados por un espacio indicando respectivamente la ganancia o pérdida de par e impar, proveniente de aplica el método de Labouchere inverso a las apuestas de par e impar en la serie de bolas contenidas en la entrada.

### EJEMPLO

ENTRADA	SALIDA
8 31 22 26 11 4 36 0 4	16 -25

## SECUENCIAS ESTABLES

### DESCRIPCIÓN

Se tiene una secuencia de números enteros  $S = \{s_1, s_2, \dots, s_N\}$ . Los números de la secuencia tienen valores entre 0 y 5000. Podemos transformar la secuencia  $S$  en otra secuencia  $T = \{t_1, t_2, \dots, t_N\}$  de la siguiente forma: se escoge uno de los números  $s_i$  de la secuencia  $S$ , se cuenta cuantas veces aparece  $s_i$  en la secuencia  $S$ , (digamos  $k$  veces) se reemplazan todas las ocurrencias del número  $s_i$  por el número  $k$  y todos los demás enteros de la secuencia se dejan igual. Por ejemplo, si comenzamos con la secuencia  $S = \{3, 1, 4, 1, 5, 9, 2\}$  y escogemos el número 1 que aparece 2 veces obtenemos la secuencia del número la secuencia  $T = \{3, 2, 4, 2, 5, 9, 2\}$ .

En el caso de que cualquier transformación posible de  $S$  produzca la misma secuencia  $S$  decimos que  $S$  es una secuencia estable. Por ejemplo, la secuencia  $S = \{3, 1, 4, 1, 5, 9, 2\}$  no es estable pues una de las transformaciones posibles produce  $T = \{3, 2, 4, 2, 5, 9, 2\}$  que es distinta a  $S$ , mientras que la secuencia  $S = \{2, 1, 2\}$  es estable.

Se sabe que comenzando con cualquier secuencia  $S$  y haciendo transformaciones de forma arbitraria siempre se llega en algún momento a una secuencia estable. Siguiendo con el ejemplo del primer párrafo, si escogemos el 2 que aparece 3 veces obtenemos la secuencia  $\{3, 3, 4, 3, 5, 9, 3\}$ , si escogemos el 9 que aparece 1 vez obtenemos  $\{3, 3, 4, 3, 5, 1, 3\}$ , si escogemos el 3 que aparece 4 veces obtenemos  $\{4, 4, 4, 4, 5, 1, 4\}$ , si escogemos el 5 que aparece una vez obtenemos  $\{4, 4, 4, 4, 1, 1, 4\}$ , si escogemos el 1 que aparece 2 veces obtenemos  $\{4, 4, 4, 4, 2, 2, 4\}$  y si escogemos el 4 que aparece 5 veces obtenemos  $\{5, 5, 5, 5, 2, 2, 5\}$ , la cual es una secuencia estable.

### PROBLEMA

Escribe un programa, que dada una secuencia de números, encuentre una serie de transformaciones tan corta como pueda, que lleve a una secuencia estable.

### ENTRADA

Tu programa deberá leer del teclado de la PC los siguientes datos.

- En la primera línea el número  $0 < N \leq 25,000$  de elementos de la secuencia  $S$ .
- En la segunda línea habrá  $N$  números enteros separados por un espacio cada uno que representan los elementos de la secuencia. Todos ellos tendrán valores entre 0 y 5000.

### SALIDA

Tu programa deberá escribir en la pantalla de la PC dos líneas. En la primera el número  $M$  de transformaciones realizadas por tu programa. En la segunda  $M$  números separados por espacios, siendo ellos los elementos seleccionados para llevar a cabo la transformación y escritos en el orden en el que se hicieron las transformaciones.

### EJEMPLO

ENTRADA	SALIDA
7	7
3 1 4 1 5 9 2	1 2 9 3 5 1 4

### FORMA DE EVALUACIÓN

En este problema, en cada caso, tu resultado será comparado contra el mejor y el peor resultado conocidos para ese caso. **El número de puntos que obtengas dependerá de que tan corta sea tu secuencia.**

## APÉNDICE DE MANEJO DE MATRICES

Una matriz es una estructura de datos que te permite representar un tablero bidimensional en la memoria de la computadora. Un tablero como este podría usarse para representar el piso de tu cuarto en el problema del asesino de la celda 5.

A continuación ponemos ejemplos de cómo definirla por si quieres utilizar este tipo de estructura.

### PASCAL

```
var
  Matriz : array [1..100,1..100] of integer; {ESTO DEFINE UN TABLERO DE 100x100}

begin
  ...
  x:=Matriz[i,j]; {ASIGNA A x EL VALOR DEL TABLERO EN LA columna i Y fila j}
  ...
  Matriz[i,j]:=z; {ASIGNA z A LA POSICIÓN EN LA columna i, fila j DEL TABLERO}
  ...
end.
```

### C/C++

```
int Matriz[100][100]; /* DEFINE UN TABLERO DE 100x100, CUYAS POSICIONES VAN DE LA
                      COLUMNA 0 A LA 99, Y DE LA FILA 0 A LA 99. LAS MATRICES EN
                      C/C++ SIEMPRE EMPIEZAN A PARTIR DE 0 */

int main(void){
  ...
  x=Matriz[j][i]; /*ASIGNA A x EL VALOR DEL TABLERO EN LA columna i Y fila j*/
  ...
  Matriz[j][i]=z; /*ASIGNA z A LA POSICIÓN EN LA columna i, fila j DEL TABLERO*/
  ...
}
```