

## Elecciones presidenciales

Las elecciones presidenciales de Estados Unidos se rigen en base a un sistema de voto indirecto. Los ciudadanos no eligen directamente al presidente, sino que votan a los miembros del llamado *Colegio Electoral*, formado por 538 compromisarios, y estos últimos son los que votan a los candidatos presidenciales. Cada uno de los estados de EEUU tiene un número determinado de compromisarios en el Colegio Electoral. Por ejemplo, el estado de Arizona tiene 11 compromisarios, el de Kansas tiene 6, el de Georgia tiene 16, etc.

Dentro de cada estado, el partido que obtiene la mayoría de votos de los ciudadanos se lleva *todos* los compromisarios de ese estado, aunque haya obtenido la mayoría por un estrecho margen<sup>1</sup>. Por ejemplo, en las pasadas elecciones del año 2020, el Partido Demócrata obtuvo los 16 compromisarios del estado de Georgia habiendo obtenido un 49,5% de los votos populares frente al 49,2% del Partido Republicano.

En este problema vamos a implementar un sistema de contabilización de votos para un país imaginario que tenga este mismo mecanismo electoral. Este sistema estará encapsulado en un TAD `ConteoVotos` que proporcione las siguientes operaciones:

- `void nuevo_estado(const string &nombre, int num_compromisarios)`

Registra un nuevo estado en el sistema, con el nombre y número de compromisarios pasados como parámetro. Si ya existía un estado registrado con el mismo nombre, se lanzará una excepción de tipo `std::domain_error` con el mensaje `Estado ya existente`. Puedes suponer que `num_compromisarios > 0`.

- `void sumar_votos(const string &estado, const string &partido, int num_votos)`

Suma `num_votos` a la cantidad de votos recibida por el partido dentro del estado indicado como parámetro. Si el estado no estaba registrado previamente en el sistema, se lanzará una excepción `std::domain_error` con el mensaje `Estado no encontrado`. Puedes suponer que `num_votos > 0`.

- `string ganador_en(const string &estado) const`

Devuelve el nombre del partido que más votos ha obtenido en el estado indicado. Puedes suponer (sin necesidad de comprobarlo en tu código) que el estado ha recibido previamente al menos un voto para algún partido político, y que el partido mayoritario tiene un número de votos estrictamente mayor que los partidos restantes; es decir, no hay empates. Si el estado indicado no estaba registrado previamente en el sistema, se lanzará una excepción `std::domain_error` con el mensaje `Estado no encontrado`.

- `vector<pair<string,int>> resultados() const`

Devuelve una lista de los partidos que han obtenido al menos un compromisario en alguno de los estados. Cada elemento de la lista es un par (`nombre`, `num_total`), donde `nombre` es el nombre del partido, y `num_total` es la suma total de compromisarios que ha obtenido en todo el país. La lista debe estar ordenada alfabéticamente según el nombre del partido. Puedes suponer (sin necesidad de comprobarlo en el código) que todos los estados registrados en el sistema tienen un partido ganador, es decir, que no hay empate en ninguno de ellos.

---

<sup>1</sup>Hay dos excepciones a este método *winner-take-all*, que son los estados de Nebraska y Maine, pero en este problema supondremos que todos los estados se rigen por este sistema.

Se pide:

1. Implementar las operaciones del TAD descrito, utilizando las librerías de la STL. Ninguno de los métodos del TAD debe realizar operaciones de E/S. El manejo de E/S debe hacerse de manera externa al TAD (en la función `resuelveCaso()`).
2. Indicar el coste de cada una de las operaciones.

## Entrada

La entrada consta de una serie de casos de prueba. Cada caso está formado por una serie de líneas, en las que se muestran las operaciones a llevar a cabo, una por cada línea: el nombre de la operación seguido de sus argumentos. La palabra `FIN` en una línea indica el final de cada caso.

Todos los nombres de estados y partidos políticos son secuencias de caracteres alfanuméricos sin espacios.

## Salida

Las operaciones `nuevo_estado` y `sumar_votos` no producen salida, salvo en caso de error. Con respecto a las restantes:

- Tras llamar a `ganador_en` debe escribirse la cadena `Ganador en XXX: ZZZ`, donde `XXX` es el estado pasado como parámetro, y `ZZZ` es el resultado devuelto por la operación.
- Tras llamar a `resultados` deben imprimirse tantas líneas como elementos tenga la lista que haya sido devuelta por esta operación. Cada línea debe contener el nombre del partido político y el número de compromisarios total, separados por un espacio.

Cada caso termina con una línea con tres guiones (`---`). Si una operación produce una excepción, debe escribirse una línea con el mensaje de la excepción, y no escribirse nada más para esa operación.

## Entrada de ejemplo

```
nuevo_estado Arizona 11
nuevo_estado RhodeIsland 4
nuevo_estado California 55
sumar_votos Arizona REP 1000
sumar_votos Arizona DEM 500
ganador_en Arizona
sumar_votos California DEM 1000
sumar_votos RhodeIsland REP 200
ganador_en California
sumar_votos Arizona DEM 600
ganador_en Arizona
resultados
FIN
nuevo_estado Nevada 6
sumar_votos Arizona P1 10
sumar_votos Nevada P2 200
ganador_en Nevada
ganador_en Arizona
FIN
nuevo_estado Illinois 20
sumar_votos Illinois P1 10
ganador_en Illinois
resultados
sumar_votos Illinois P2 20
ganador_en Illinois
resultados
sumar_votos Illinois P3 30
ganador_en Illinois
resultados
FIN
```

## Salida de ejemplo

```
Ganador en Arizona: REP
Ganador en California: DEM
Ganador en Arizona: DEM
DEM 66
REP 4
---
Estado no encontrado
Ganador en Nevada: P2
Estado no encontrado
---
Ganador en Illinois: P1
P1 20
Ganador en Illinois: P2
P2 20
Ganador en Illinois: P3
P3 20
---
```

## Autor

Manuel Montenegro