UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA
Programación II – Curso 2022/23

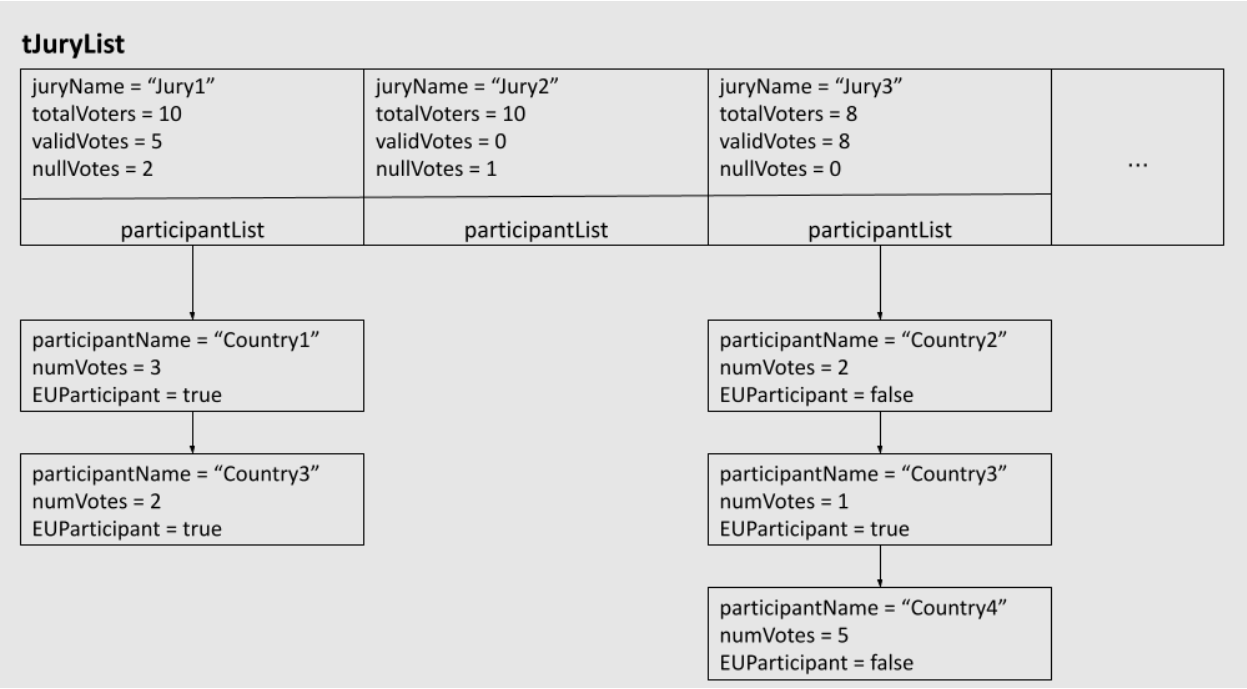
Práctica 2: Enunciado

1. El problema

En esta segunda práctica se amplían las funcionalidades de la primera, pasando ahora a gestionarse simultáneamente los resultados de las votaciones de varios jurados locales. El sistema se encargará de llevar un recuento de votos actualizado de todos los jurados, pudiendo mostrar además estadísticas de los mismos. Para conseguirlo, se usarán distintas estructuras de datos.

El objetivo de esta práctica es comprender el funcionamiento e implementar varios tipos abstractos de datos (TADs), así como manejar interdependencias entre ellos.

2. Fase 1



Esta primera fase se centrará en los tipos abstractos de datos (TADs). Para resolver este problema se utilizarán 2 TADs:

- Una Lista Ordenada (TAD ParticipantList) para almacenar las votaciones a los participantes correspondientes a un único jurado.
- Una Lista Ordenada (TAD JuryList) para almacenar los distintos jurados.

Cada jurado, esto es, cada nodo de la lista de jurados, tendrá su propia lista de participantes, tal y como se muestra en la figura anterior.

2.1. Librería Types

Algunos tipos de datos comunes se definirán en la librería types.h, ya que son utilizados simultáneamente en varios TADs.

`tJuryName`	Nombre de un jurado (`string`).
`tParticipantName`	Nombre de un participante (`string`).
`tNumVotes`	Número de votos (`int`).
`tEUParticipant`	Pertenencia a la Unión Europea (`bool`).

2.2. TAD ParticipantList

Se realizará una implementación **dinámica y simplemente enlazada** del TAD Lista **Ordenada** (`participant_list.h y participant_list.c`) para almacenar una lista de participantes.

2.2.1. Tipos de datos incluidos en el TAD ParticipantList

`tListP`	Representa una lista de participantes.
`tItemP`	Datos de un elemento de la lista (un participante). Compuesto por: • `participantName`: de tipo `tParticipantName` • `numVotes`: de tipo `tNumVotes` • `EUParticipant`: de tipo `tEUParticipant`
`tPosP`	Posición de un elemento en la lista de participantes.
`NULLP`	Constante utilizada para representar posiciones nulas en la lista de participantes.

2.2.2. Operaciones incluidas en el TAD ParticipantList

Las operaciones son las mismas que las del TAD Lista de la Práctica 1 renombrando los tipos de datos y las operaciones para evitar conflictos con el TAD JuryList. Todos los identificadores

del TAD ParticipantList pasarán a terminar en P. Cambia la especificación de las siguientes operaciones:

- `insertItemP(tItemP, tListP) → tListP, bool`
  Inserta un elemento de forma **ordenada** en función del campo `participantName`. Devuelve un valor `true` si el elemento fue insertado; `false` en caso contrario. **PostCD: Las posiciones de los elementos de la lista posteriores a la del elemento insertado pueden haber variado.**

Asimismo, la operación `findItemP` no cambia su especificación, pero su implementación debe tener en cuenta que ahora la lista está **ordenada**.

2.3. TAD JuryList

Se realizará una implementación **estática** del TAD Lista **Ordenada** (`jury_list.h` y `jury_list.c`) para mantener la lista de jurados (con capacidad máxima de 10 jurados).

2.3.1. Tipos de datos incluidos en el TAD JuryList

`tListJ`	Representa una lista de jurados.
`tItemJ`	Datos de un elemento de la lista (un jurado). Compuesto por: • `juryName`: de tipo `tJuryName` • `totalVoters`: de tipo `tNumVotes` • `validVotes`: de tipo `tNumVotes` • `nullVotes`: de tipo `tNumVotes` • `participantList`: de tipo `tListP`
`tPosJ`	Posición de un elemento en la lista de jurados.
`NULLJ`	Constante utilizada para representar posiciones nulas en la lista de jurados.

2.3.2. Operaciones incluidas en el TAD JuryList

Las operaciones son las mismas que las del TAD Lista de la Práctica 1 renombrando los tipos de datos y las operaciones para evitar conflictos con el TAD ParticipantList. Todos los identificadores del TAD JuryList pasarán a terminar en J. Cambia la especificación de las siguientes operaciones:

- `insertItemJ (tItemJ, tListJ) → tListJ, bool`
  Inserta un elemento de forma **ordenada** en función del campo `juryName`. Devuelve un valor `true` si el elemento fue insertado; `false` en caso contrario. **PostCD: Las posiciones de los elementos de la lista posteriores al insertado pueden cambiar de valor.**

- `deleteAtPositionJ (tPosJ, tListJ) → tListJ`
  Elimina de la lista el elemento que ocupa la posición indicada.

PreCD: La posición indicada es una posición válida en la lista **y el jurado en dicha posición tiene una lista de participantes vacía.**
PostCD: Las posiciones de los elementos de la lista posteriores a la de la posición eliminada pueden haber variado.

Asimismo, la implementación de la operación `findItemJ` debe tener en cuenta que ahora la lista está **ordenada** y que devuelve la posición del primer elemento de la lista **cuyo nombre de jurado** (campo `juryName`) se corresponda con el indicado (o `NULLJ` si no existe tal elemento).

- `findItemJ (tJuryName, tListJ) → tPosJ`

3. Fase 2

Una vez implementado los TDAs, nos centraremos en el programa principal. La tarea consiste ahora en implementar un programa principal (`main.c`) que **use los TAD ParticipantList y JuryList** y que procese las peticiones recibidas con el siguiente formato:

`C juryName totalVoters`	***[C]reate***: Se incorpora el jurado `juryName` cuyo número total de miembros es `totalVoters`.
`N juryName participantName EUParticipant`	***[N]ew:*** Alta de un participante en un jurado y con 0 votos. `EUParticipant` puede tomar el valor *eu* o *non-eu*.
`V juryName participantName`	***[V]ote:*** Se emite un voto para un participante en un jurado.
`D participantName`	***[D]isqualify:*** Descalifica a un participante eliminándolo de todas las listas de jurados, y pasando a ser votos nulos todos sus votos.
`R`	***[R]emove:*** Elimina los jurados con 0 votos válidos.
`S`	***[S]tats:*** Muestra estadísticas de participación y voto.
`W`	***[W]inners:*** Para cada jurado muestra el ganador entre los países que pertenecen a la Unión Europea y entre los que no.

El programa leerá y procesará las operaciones contenidas en un fichero y se ejecutarán dichas operaciones, tal y como se hizo en la Práctica 1. Para cada petición, el programa:

1. Muestra una cabecera con la operación a realizar. Esta cabecera está formada por una primera línea con 20 asteriscos y una segunda línea que indica la operación.

```
********************
CC␣T:␣jury/participant␣XX␣totalvoters/participant␣YY␣location␣ZZ
```

donde `CC` es el número de petición; `T` es el tipo de operación (`C`, `N`, `V`, `D`, `R`, `S` o `W`); `XX` es, o bien el nombre del jurado (`juryName`) o bien el nombre del participante (`participantName`). De forma similar, `YY` es, o bien el número de miembros del jurado (`totalVoters`), o bien el nombre del participante (`participantName`), Finalmente, `ZZ` indica si el participante representa a un país de la Unión Europea (*eu* o *non-eu* según el campo `EUParticipant`).

2. Procesa la petición correspondiente:

- Si la operación es **`[C]reate`**, se añade el jurado a la lista con el número de votantes igualado al parámetro indicado. Además, se mostrará:

  ```
  *␣Create:␣jury␣XX␣totalvoters␣YY
  ```

  donde `XX` es el nombre del jurado (`juryName`), `YY` es el número total de miembros de ese jurado. Si ya existiese un jurado con el nombre indicado (`juryName`) o hubiese algún problema al crear el jurado, se mostrará:

  ```
  +␣Error:␣Create␣not␣possible
  ```

- Si la operación es **`[N]ew`**, se añadirá el participante al jurado indicado y se mostrará:

  ```
  *␣New:␣jury␣XX␣participant␣YY␣location␣ZZ
  ```

  donde `XX` es el nombre del jurado, `YY` es el del participante y `ZZ` es el valor `EUParticipant` (se imprimirá *eu* o *non-eu* en función del *bool*). Si ese participante ya existiese en el jurado o hubiese algún problema para realizar la inserción, se mostrará:

  ```
  +␣Error:␣New␣not␣possible
  ```

- Si la operación es **`[V]ote`**, se incrementará el número de votos de ese participante en el jurado y se mostrará:

  ```
  *␣Vote:␣jury␣XX␣participant␣YY␣location␣ZZ␣numvotes␣VV
  ```

  donde `XX` es el nombre del jurado, `YY` el del participante, `ZZ` es el valor `EUParticipant` y `VV` es el número de votos actualizado. Si el jurado no existe, se muestra:

  ```
  +␣Error:␣Vote␣not␣possible
  ```

  Si el participante en cuestión no existiese para el jurado indicado, ese voto se contabilizará como nulo y se mostrará:

  ```
  +␣Error:␣Vote␣not␣possible.␣Participant␣YY␣not␣found␣in␣jury␣XX.␣NULLVOTE
  ```

- Si la operación es **`[D]isqualify`**, se buscará al participante en todas las listas locales, se borrará de las listas, pasando sus votos a considerarse como NULO y se imprimirá el siguiente mensaje:

```
Jury␣CC1
Participant␣XX␣disqualified

Jury␣CC2
       …
```

donde separamos un jurado de otro mediante una línea en blanco.

Si la lista de jurados está vacía, se imprimirá el siguiente mensaje:

```
+␣Error:␣Disqualify␣not␣possible
```

Si no existiese el participante `XX` en alguna de las listas locales, se indicaría tal que así:

```
Jury␣CC1
No␣participant␣XX

Jury␣CC2
       …
```

- Si la operación es **`[R]emove`**, se eliminarán todos los jurados con 0 votos válidos, mostrando el siguiente mensaje para cada uno de ellos:

```
*␣Remove:␣jury␣XX
```

Si no existiese ningún jurado a eliminar o si la lista de jurados está vacía, se mostrará:

```
+␣Error:␣Remove␣not␣possible
```

- Si la operación es **`[S]tats`**, se mostrarán las estadísticas de voto de la siguiente forma:

```
Jury␣CC1
Participant␣XX1␣location␣LL1␣numvotes␣YY1␣(ZZ1%)
…
Participant␣XXn␣location␣LLn␣numvotes␣YYn␣(ZZn%)
Nullvotes NN
Participation:␣KK␣votes␣from␣VV␣voters␣(PP%)

Jury␣CC2
         …
```

donde `CC` es el nombre del jurado y el resto de estadísticas se muestran de la misma forma que en la Práctica 1. Obsérvese que, de nuevo, separamos un jurado de otro mediante una línea en blanco. Asimismo, si la lista de jurados está completamente vacía, se imprimirá el mensaje:

```
+␣Error:␣Stats␣not␣possible
```

mientras que si es la lista de participantes de algún jurado está vacía, esto se indicará tal que así:

```
Jury␣CC1
No participants
Nullvotes NN
Participation:␣KK␣votes␣from␣VV␣voters␣(PP%)

Jury␣CC2
        …
```

- Si la operación es **`[W]inners`**, se mostrarán los ganadores de cada jurado en función de si pertenecen o no a la Unión Europea:

  ```
  Jury␣CC1
  Location␣eu: Participant␣XX1␣numvotes␣YY1
  Location␣non-eu: Participant␣XX2␣numvotes␣YY2

  Jury␣CC2
         …
  ```

  donde, de nuevo, separamos un jurado de otro mediante una línea en blanco. Si en una localización hubiese un único participante con 0 votos, dicho participante se considerará el ganador.

  Si no existiese ganador, hubiese empate o no hubiese participantes con esa localización (*eu* o *non-eu*) se mostrará:

  ```
  Jury␣CC1
  Location␣eu: No winner
  Location␣non-eu: No winner

  Jury␣CC2
         …
  ```

  Asimismo, si la lista de jurados está completamente vacía, se imprimirá el mensaje:

  ```
  +␣Error:␣Winners␣not␣possible
  ```

4. Lectura de ficheros

Para facilitar el desarrollo de la práctica se proporciona el siguiente material de especial interés: (1) Un directorio `CLion` que incluye un proyecto plantilla (`P2.zip`) y, (2) Un directorio `script` donde se proporciona un fichero (`script.sh`) que permite probar de manera conjunta los distintos archivos proporcionados. Además, se facilita un documento de ayuda para su ejecución (`EjecucionScript_P2.pdf`). Nótese que, para que el `script` no dé problemas se recomienda **NO copiar directamente el código de este documento**, ya que el formato PDF puede incluir caracteres invisibles que darían por incorrectas salidas (aparentemente) válidas.

5. Información Importante

El documento `NormasEntrega.pdf`, disponible en la página web de la asignatura detalla claramente las normas de entrega. Para un mejor **seguimiento de la práctica** se realizarán dos **entregas parciales obligatorias** antes de las fechas y con los contenidos que se indican a continuación:

- **Entrega parcial #1: viernes 14 de abril a las 22:00 horas.** Implementación y prueba del TAD Participant List y el TAD Jury List (entrega de los ficheros `types.h`, `participant_list.c`, `participant_list.h`, `jury_list.c` y `jury_list.h`).
- **Entrega parcial #2: viernes 21 de abril a las 22:00 horas.** Implementación y prueba de las siguientes funcionalidades del programa principal: `Create`, `New`, `Stats` y `Vote` (entrega de los ficheros `types.h`, `participant_list.c`, `participant_list.h`, `jury_list.c`, `jury_list.h` y `main.c`). Para comprobar el correcto funcionamiento de las operaciones se facilitarán los ficheros de prueba `create.txt`, `new.txt` y `vote.txt`.

Se realizará una corrección automática usando el *script* proporcionado para ver si se superan o no los correspondientes requisitos (véase documento `CriteriosEvaluacion.pdf`).

Fecha límite de entrega: **viernes 28 de abril a las 22:00 horas.**