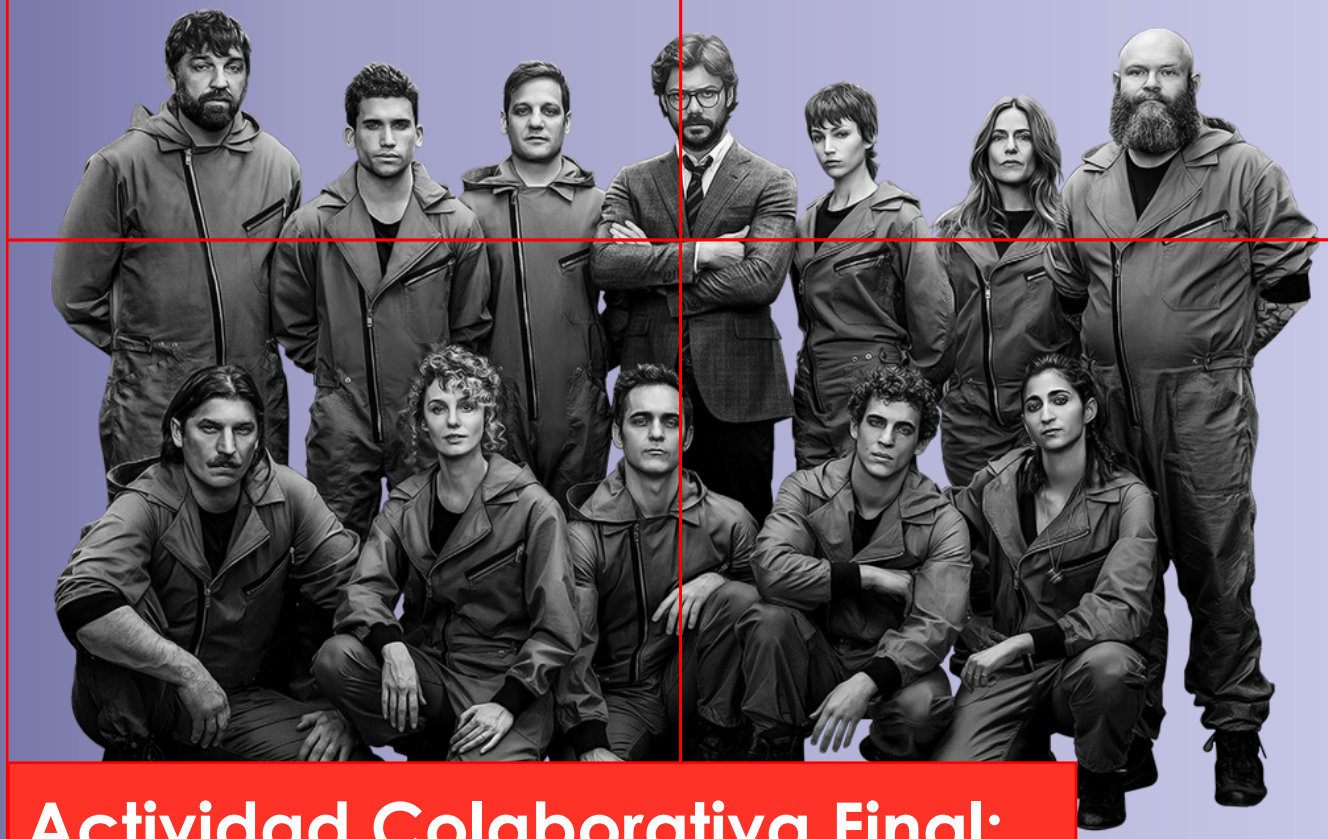
Universidad Europea



Actividad Colaborativa Final: La casa de Papel

Programación con estructuras lineales

Grado en Ingeniería Informática

Jorge García

Universidad Europea

Buenas,

Os habla El Profesor. Estoy buscando nuevos miembros para La Banda, y creo que vosotros sois buenos candidatos. Los detalles del golpe son algo… "delicados", por lo que no puedo compartir más detalles, pero para este golpe, necesitamos los mejores informáticos.

Río ha dejado preparados una serie de problemas algorítmicos, de modo que, para encontrar al mejor equipo de avanzadilla, cada uno de vosotros debe buscar a otros 5 compañeros (hasta ser un máximo de 6), con los que poder resolver estos problemas. Pero si lo preferís, también podéis intentarlo por vuestra cuenta.

Una vez que hayáis decidido los compañeros con los que vais a trabajar, tendréis que decidir un nombre en clave para la operación. Cada equipo tendrá el suyo, de modo que poneros de acuerdo, y mandadme los detalles tanto de los miembros como los del nombre. Sería inteligente por vuestra parte, que designéis un representante que mande estos detalles. Si me tengo que poner en contacto con vosotros lo haré a través de dicha persona.

Antes de plantearos los problemas, tengo que dejar claras una serie de cosas:

1. La entrega de los archivos se realizará mediante el campus virtual de la asignatura de PEL, en formato .zip, y bajo el formato de nombre: [nombre_clave_grupo]_LCDP.zip
2. Dentro del fichero a entregar, debe encontrarse una carpeta por cada uno de los problemas planteados por Río, y dentro de cada carpeta la solución a los problemas.
3. Además, deberá redactarse un documento en el que se contemplen las decisiones, y procedimientos realizados para resolver dichos problemas.

Os deseo mucha suerte en esta tarea, y que, como habría dicho mi amiga Tokio, seáis como Mozart, pero con los ordenadores.

- El Profesor

A ver chavales, aquí Río. El Profesor me ha pedido que os prepare una serie de pruebas para evaluar vuestra entrada al equipo de informáticos de la banda. Los indios no son malos, pero necesitamos a especialistas de Elite, así que cuento con vosotros.

Las pruebas son del más alto nivel y las he dejado preparadas para que demostréis que sabéis salir de las situaciones más enrevesadas. No os confiéis y hacer lo que sabéis. Dar jarana de la buena.

Así que chavales, animo y al toro.

Situación 1:

A la hora de acceder a la caja fuerte de un ruso de estos ricachones, en donde están las claves de acceso a un bunker blindado en donde se guardan uno de los botines más jugosos del mundo (o eso aseguraba Berlín), nos encontramos que el patrón de ordenación de las claves de acceso sigue una secuencia de 10 números escrita en una tabla hash. Para extraer los datos, debemos ordenar los elementos de tipo *double* de esa tabla hash de mayor a menor.

Tenéis que diseñar un objeto llamado *deque*, en el que podáis almacenar los elementos ubicados en las tablas hash. Tras esto, implementad un algoritmo de ordenación (podéis llamarlo *sort*) que ordene la secuencia de dichos elementos, pudiendo así obtener las claves de desbloqueo de la caja fuerte del ruso.

Situación 2:

Bien chavales, en esta situación tenéis que plantearos que hay que hacer también reconocimiento de la gente que hay dentro de la seguridad de la casa del ruso. Para poder categorizar a los equipos de seguridad, se necesita un sistema que permita recoger la información de los seguratas que tiene el ruso en su casa.

El programa almacenará la información del nombre del segurata y un número correspondiente a la puntuación que sacaron en su evaluación como candidatos para securizar la casa del ruso.

Como de primeras no sabemos el número de seguratas que hay, el programa debe permitir introducir tantos seguratas como sea posible (usar para ello la clase std::vector).

Una vez introducidos los datos de todos los seguratas, el sistema debe ordenarlos de mayor a menor puntuación para saber quienes son los guardias con los que más cuidado debemos de tener, mostrando la información ordenada por pantalla.

Situación 3:

Bien, pasamos a la situación 3. Esta vez queremos poder manejar la información almacenada en el vector de seguratas de la situación anterior. Planteando que el TAD InfoSegurata tiene la siguiente estructura, hay que realizar lo siguiente:

```
struct InfoSegurata
{
    string nombre_segurata;
    int evaluacion_candidato;
};
```

- Se siguen almacenando los datos dentro de un objeto std::vector<InfoSegurata> y debe estar ordenado por la evaluación del candidato.
- Es necesario realizar un programa, que pueda calcular y obtener las evaluaciones máximas y mínimas de cada segurata. Así podremos saber cual es el segurata más y menos peligroso.
- También se debe obtener el promedio de las seguratas para saber la dificultad de este y futuros golpes.
- Toda esta información debe mostrarse en pantalla.

Situación 4:

Bien chavales, esta va a ser la última situación. Siguiendo los desarrollos realizados hasta ahora. Deberéis escribir una función que separe en 2 los datos almacenados en el vector. En el primero estarán únicamente aquellos más peligrosos (una evaluación mayor a 8), y en el otro aquellos que, dentro de ser chungos, no van a dar muchos problemas.

Pero, en este caso tenéis que hacerlo de dos formas diferentes:

- La primera forma es usando 2 vectores adicionales, uno de seguratas peligrosos y otro de seguratas menos peligrosos, conservando el original durante la separación y eliminándolo al terminar de separarlos.
- La otra forma, es generar una función de seguratas peligrosos, que compruebe del vector original cuales son peligrosos y sacándolos del vector original e introduciéndolos en el nuevo. Dejando en el vector original los menos peligrosos y en el nuevo los más peligrosos.
- Para ambas formas, realizar las funciones sin usar métodos iterativos. Únicamente recursivos.

Universidad Europea

Escuela STEAM

Jorge García González
jorge.garcia@universidadeuropea.es
Despacho C32

Ve más allá