PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada
2^o semestre 2016

Actividad 05

Programación Funcional

Instrucciones

Dedicarte a ser pokemon trainer y dejar de lado los estudios te ha jugado una mala pasada. Por un lado, tus notas te hacen querer llorar [1]. Peor aún, no entiendes porque la gente ha dejado de jugar de manera tan apasionada tu amado juego y tu valiosa colección de pokemones está en peligro de morir con la extinción de Go. Tal como Ash, decides no rendirte y no dejar de lado tu vida de entrenador Pokemon. Tomas la decisión de hacer que Pokemon Go vuelva a retomar el camino de la gloria.

Para lograr esto, aprovechando que te has convertido en un gran programador[2], has decidido organizar un torneo pokemón. Trás una gran campaña de marketing has logrado llegar a una larga lista de inscritos. Esta lista es tan larga que ya no sabes como organizar el torneo. Buscando alternativas, encuentras el sistema suizo.

El Sistema Suizo es un método para organizar torneos o enfrentamiento de partidas entre 2 jugadores. Dada la lista de jugadores con su ranking inicial (elo), el formato y reglas del torneo son:

- En la primera ronda, se ordenan los jugadores según su elo (el cual se encuentra en el archivo jugadores.csv) y se emparejan de a dos, el participante j con el j + 1. Los miembros de cada pareja (de esta ronda) deberán pelear uno contra el otro. Cabe destacar que el numero total de participantes es par.
- Todos los partidos tienen un ganador y un perdedor. El ganador se le otorgan 6 puntos que se acumulan y una partida ganada, mientras que el perdedor recibe un puntaje aleatorio menor a 6 y ninguna partida ganada.
- En las rondas posteriores, se forman grupos según la cantidad de victorias (partidas ganadas, no por puntaje ni por elo). Estos grupos son ordenados según el puntaje acumulado y luego se emparejan de a dos de la misma forma que en la primera ronda. En caso de haber un número impar de personas en el grupo, el último del grupo deberá pelear con el mejor del grupo de una victoria menos.
- Este proceso se repite continuamente hasta llegar a las 10 rondas, el ganador es aquel que haya ganado más partidas que nadie.

Lamentablemente, no tienes presupuesto para un torneo de más de 10 rondas, por lo que **no necesariamente existe un ganador**.

[1] Excepto en progra que era lo esperado
[2] O lo serás a fin de semestre

Paralelo al torneo, has decidido crear un sistema de apuestas de quien será el ganador en donde solamente participan los jugadores del torneo. Como tu cortarás el 50 % de estas apuestas para recuperar los costos del torneo[3], necesitas saber el monto acumulado que se aposto durante el final de cada ronda el cual varía de 0 a 1000.

Requerimientos

- Deberás leer el archivo .csv de jugadores, de tal forma en que solo se utilizen generadores, hasta la creación de los jugadores (hint: jugador será una clase), donde puedes utilizar una lista.
- Tu programa debe ser completamente funcional. No se podrá utilizar for ni while. While SOLO puede ser utilizado dentro de generadores. Puedes usar for dentro de una lista o generador de comprensión.
- Deberás modelar el campeonato con sus participantes usando las clases y métodos apropiados.
- Cada jugador debe tener un id, que debe ser creado con un generador.
- Simular la realización del torneo y sus 10 rondas.
- Implementar el sistema de apuestas de tal manera en que las apuestas se realizen con un **generador**.

PROHIBIDO/PERMITIDO

```
While prohibido:
        while i > k:
                do something

While permitido:
        def generador():
                while i > k:
                        yield x
                        do something

For prohibido:
        lista = [1,2,3,4,5]
        def funcion(valor):
                return valor + 10
        for j in lista:
                funcion(j)

For permitido:
        lista = [1,2,3,4,5]
        def funcion(valor):
                return valor + 10
        lista\_nueva = [funcion(j) for j in lista]
```

Pero lo último se podría haber hecho con un ... [4]

[3] ¡todos ganan!

[4] Es una de las funciones clásicas de programaión funcional ;)

To - DO

- (1.50 pts) Correcto uso de funcional
- (1.00 pts) Correcta lectura del archivo de texto
- (1.00 pts) Simulación exitosa del torneo
- (0.75 pts) Correcto sistema de apuestas y creación de id's.
- (0.5 pts) Correcta modelación del problema

Debes imprimir en pantalla las siguientes consultas:

- (0.25 pts) Monto acumulado de apuestas al final de cada ronda
- (0.50 pts) Promedio de los promedios de puntajes de cada jugador por grupo
- (0.50 pts) Lista ordenada por puntaje promedio y nombre del jugador del grupo perdedor (menor puntaje)

Bonus

Si esta actividad te ha parecido muy fácil te presentamos el siguiente desafio:

- (0.5 pts) Promedio del puntaje promedio en partidos perdidos por jugadores que han ganado menos de 5 partidas.
- (0.5 pts) No utilizar ningún for de comprensión en nninguna parte del código.

Entrega

- **Lugar:** GIT - Carpeta: Actividades/AC05
- **Hora:** 16:55