Anotador de tenis

Pensamiento Computacional

Trabajo Práctico N° 1

1. Objetivos

En este trabajo práctico se busca implementar un programa con la capacidad de manejar el indicador de puntaje de cada jugador durante un *game* de tenis. El programa debe tener dos modos:

1. Modo manual: debe permitir que el usuario ingrese el ganador de cada punto.
2. Modo simulación: debe poder simular el ganador del punto.

2. Alcance del Trabajo Práctico

Temas que comprende este trabajo práctico:

- Variable y tipos
- Funciones
- Condicionales y ciclos
- Secuencias
- Diagramas de Flujo

3. Introducción

El tenis es un deporte que utiliza un sistema de puntuación del estilo "ida y vuelta". Este tipo de puntuación resulta menos intuitiva si se la compara con otros deportes o juegos en donde se contabilizan los puntos o aciertos de cada equipo como un simple contador de números naturales positivos. En relación a esto, el trabajo práctico estará basado en la implementación de un indicador de puntaje durante un game de tenis. Para ello consideremos las reglas que se resumen a continuación.

Reglas del tenis

Un partido de tenis está compuesto por *sets*, tal que el jugador que logre ganar en 3 o 5 sets de manera irremontable, será el ganador del partido. A su vez, cada set está integrado por distintos *games*, de modo

que el jugador que alcance primero los 6 games (o más, dependiendo de la diferencia con su oponente), gana el set. Por su parte, cada game consta de puntos que se ganan en base a las reglas que se detallan a continuación:

1. Ambos jugadores comienzan cada nuevo game con puntaje **0**.
2. Cada vez que un jugador anota un punto dentro del game, se incrementa su puntaje según la secuencia: **0** → **15** → **30** → **40**.
3. Si un jugador posee **40** puntos y gana un nuevo punto, ganará el game en caso que su oponente posea un puntaje inferior a **40**.
4. Si ambos jugadores poseen **40** puntos sin una ventaja previa de ninguno de los dos, el jugador que gane de nuevo punto tendrá **ventaja**.
5. Si un jugador posee **ventaja** y pierde el siguiente punto, vuelven a estar iguales **40 – 40**, sin ventaja para ninguno.
6. Si un jugador posee **ventaja** y gana el nuevo punto, gana el game.

4. Desarrollo del TP

4.1. Requerimientos

De acuerdo a los objetivos definidos inicialmente para los dos modos de funcionamiento, se establecen los siguientes requerimientos:

Modo manual

- Inicialmente, se debe poder ingresar el nombre de ambos jugadores.
- En cada punto jugado se debe poder ingresar el ganador del punto (identificado con el nombre ingresado inicialmente).
- La respuesta del programa luego haber ingresado el ganador del punto debe ser el nombre de cada jugador y el puntaje actual.
- Cuando uno de los jugadores esté en **ventaja**, el puntaje indicado debe ser *Ad*.
- Al finalizar debe indicarse quién fue el ganador del game.

> *Aclaración:* notar que se pide la implementación para un game, y no para un set o partido completo.

Modo simulación

- Inicialmente, se debe poder ingresar el nombre de ambos jugadores.
- Se debe poder elegir si se quiere visualizar un detalle de todo el game (con el mismo formato que el *Modo manual*) o sólo quien ganó.
- La máquina debe generar al azar el ganador de cada punto e indicar el nombre luego de `Who scored?`.

> *Aclaración:* notar que se pide la implementación para un game, y no para un set o partido completo.

4.2. Diagrama de flujo

En primer lugar plantee una solución al problema mediante un diagrama de flujo para que el programa cumpla los requerimientos para luego ser plasmado en un script. Analice las entradas de datos necesarias,

qué procesos deben realizarse y las salidas que se deben obtener en cada etapa. En el Apéndice se agregan ejemplos de cómo debería responder el programa.

4.3. Codificación

Luego de concluir el diseño del programa mediante el diagrama de flujos, se deberá codificar el algoritmo en python y verificar el cumplimiento de los requerimientos y eventualmente realizar correcciones. El código debe estar debidamente documentado, entre otras cosas, todas las funciones deben tener sus *docstrings*. Cualquier simplificación o abreviatura de la nomenclatura entre la notación del diagrama de flujo y las variables o funciones del código deben ser aclaradas.

5. Uso

A continuación se muestran ejemplos de lo que debería verse en pantalla, en modo *manual* o *simulación*, para la aplicación a desarrollar.

Ejemplo: inicio

```
** Welcome to the Super Tennis Annotator app **
How do you want to use it?
1. Manual input
2. Simulation
>
```

Ejemplo: modo manual

```
** Super Tennis Annotator **
Insert player 1's name: Ada
Insert player 2's name: Alan
Score:
Ada    0
Alan   0
Who scored? Alan
Score:
Ada    0
Alan   15
Who scored? Ada
Score:
Ada    15
Alan   15
Who scored? Ada
Score:
Ada    30
Alan   15
Who scored? Ada
Score:
Ada    40
Alan   15
Who scored? Alan
```

```
Score:
Ada    40
Alan   30
Who scored? Ada
Score:
Ada wins the game.
```

Ejemplo: modo simulación (automático)

```
** Super Tennis Annotator **
Insert player 1's name: Ada
Insert player 2's name: Alan
Score:
Ada    0
Alan   0
Who scored? Alan
Score:
Ada    0
Alan   15
Who scored? Ada
Score:
Ada    15
Alan   15
Who scored? Ada
Score:
Ada    30
Alan   15
Who scored? Ada
Score:
Ada    40
Alan   15
Who scored? Alan
Score:
Ada    40
Alan   30
Who scored? Alan
Score:
Ada    40
Alan   40
Who scored? Alan
Score:
Ada    40
Alan   Ad
Who scored? Ada
Score:
Ada    40
Alan   40
Who scored? Ada
Score:
Ada    Ad
Alan   40
```

```
Who scored? Alan
Score:
Ada    40
Alan   40
Who scored? Alan
Score:
Ada    40
Alan   Ad
Who scored? Ada
Score:
Ada    40
Alan   40
Who scored? Ada
Score:
Ada    Ad
Alan   40
Who scored? Alan
Score:
Ada    40
Alan   40
Who scored? Alan
Score:
Ada    40
Alan   Ad
Who scored? Alan
Score:
Alan wins the game
```

6. Entrega

El trabajo práctico es **individual**.

Se debe realizar una entrega digital a través del campus de la materia, compuesta por un único archivo comprimido con todos los contenidos del trabajo.

El nombre de dicho archivo debe cumplir el siguiente formato:

```
tp1_apellido.zip
```

donde `apellido` es su apellido y `.zip` es la extensión del archivo comprimido (que puede ser `.zip`, `.tar`, `.gz`, o `.tar.gz`).

A su vez, el archivo comprimido debe contener los siguientes elementos:

- La documentación del desarrollo **en formato pdf**, que contiene los siguientes items:

    1. Carátula del trabajo práctico, nombre y apellido del alumno y dirección de correo electrónico.
    2. Objetivos del trabajo.
    3. Desarrollo del trabajo.
    4. Explicación de alternativas consideradas y estrategias adoptadas.

5. Resultados de ejecuciones, incluyendo capturas de pantalla (como imagen o texto), bajo condiciones normales e inesperadas de entrada.
6. Reseña sobre problemas encontrados y soluciones.
7. Bibliografía.
8. Indicaciones para ejecutar correctamente el programa y las pruebas.

Nota: el informe debe estar redactado en *correcto* castellano.

- Código fuente.