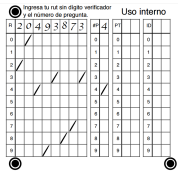
Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC1103 – Introducción a la Programación
2 - 2016

Interrogación 1

Instrucciones generales:

- La interrogación consta de tres preguntas con igual ponderación.
- Solo se reciben consultas sobre el enunciado en los primeros 30 minutos.
- La última página es un recordatorio; no está permitido utilizar material de apoyo adicional.
- Las tres preguntas están pensadas para ser resueltas en forma sencilla con ciclos y condicionales. No es necesario utilizar otros mecanismos más avanzados.
- Si el enunciado de la pregunta no lo indica explícitamente, no es necesario validar los ingresos del usuario. Puedes suponer que no hay errores por ingreso de datos.
- Responde cada pregunta en una hoja separada, marcando tu RUT y el número de pregunta en todas las hojas prestando atención a la dirección del trazo, tal como se muestra en el siguiente ejemplo:



Pregunta 1 (1/3)

Utilizando el módulo predefinido `olimpiadas`, descrito a continuación, escribe un programa que simule el funcionamiento interactivo de un programa que ofrece una serie de eventos durante un día de las olimpiadas, permite al usuario seleccionar a cuáles asistir, y en base a ello, presenta un calendario de actividades para dicho día.
Las siguientes funciones están ya implementadas en el módulo olimpiadas; debes utilizarlas para resolver este problema.

- `leer_horarios()` – lee los horarios correspondientes al día actual desde internet. Para cada actividad, carga el horario de inicio, el nombre del deporte, y la duración del evento. Esta función se debe llamar primero, para que las siguientes retornen valores adecuados.
- `cantidad_actividades()` – Retorna la cantidad de actividades que habrá en el día actual.
- `hora_inicio(id)` – Retorna un string con la hora de inicio de la actividad `id` (valores se inician en 1), en el formato hh:mm (ejemplos: `"08:00"` o `"14:00"`).
- `duracion(id)` – Retorna un int con la duración, en minutos, de la actividad `id`.
- `deporte(id)` – Retorna un string con el nombre del deporte de la actividad `id`.
- `seleccionar_actividad(id)` – Almacena que se ha seleccionado la actividad `id`. Se pueden almacenar todas las actividades que se desee. Por ejemplo, `seleccionar_actividad(5)` almacena que se seleccionó la actividad 5, y si luego se ejecuta `seleccionar_actividad(3)` han quedado almacenadas las actividades 5 y 3.
- `cantidad_seleccionadas()` – Retorna la cantidad de actividades seleccionadas.

- `seleccionada(i)` – Retorna el id de la actividad seleccionada `i`-ésima (se retornan en orden creciente, partiendo desde `i = 0`). Por ejemplo, si se seleccionaron las actividades 8, 3 y 6, `seleccionada(0)` retorna 3, `seleccionada(1)` retorna 6 y `seleccionada(2)` entrega 8.

(a) (10 puntos) Define una función `hay_traslape(h1, d1, h2, d2)` que entregue `True` si la actividad que comienza a la hora `h1` y dura `d1` minutos, se traslapa con la actividad que empieza a la hora `h2` y dura `d2` minutos. Puedes suponer que `h1` siempre es menor o igual a `h2`. Ejemplos:

- `hay_traslape("08:00", 40, "08:45", 10)` entrega `False`
- `hay_traslape("08:00", 45, "08:45", 10)` entrega `False`
- `hay_traslape("08:00", 50, "08:45", 10)` entrega `True`
- `hay_traslape("14:00", 125, "15:00", 50)` entrega `True`

(b) (20 puntos) Escribe un programa principal que lea la programación del día y la despliegue en pantalla para que la vea el usuario. Se debe mostrar un identificador, la hora de inicio, duración, y nombre del deporte. Ejemplo:

```
Las actividades de los J.J.O.O. hoy son:
1 08:00(70) Tiro con arco
2 08:00(15) Tiro skeet
3 09:05(60) Atletismo
4 09:25(200) Golf
5 11:20(25) Remo
6 11:30(60) Halterofilia
7 11:30(25) Remo
8 12:05(40) Vela
9 12:05(40) Vela
10 12:05(40) Vela
11 12:15(40) Vela
12 12:15(40) Vela
13 12:30(100) Atletismo
14 12:40(60) Natacion
15 21:00(75) Atletismo
```

(c) (30 puntos) Continúa el programa anterior de la siguiente manera:

- Debes pedirle al usuario que indique a qué actividades quiere asistir
- Tu programa debe desplegar la programación de actividades que ha escogido el usuario. Si hay traslapes de horario, tu programa debe mostrar la primera actividad a la que se asistirá y debe saltarse las que tienen traslape con ésta.
- Para esta pregunta puedes suponer que está implementada la funcion `hay_traslape`

Ejemplo:

```
Escoge los eventos a los cuales quieres asistir (0 para terminar).
1
5
3
13
2
0

Esta es tu programación:
08:00 Tiro con arco
11:20 Remo
12:30 Atletismo
```

Pregunta 2 (1/3)

Decimos que un string `s1` es substring de `s2` cuando `s1` está contenido en `s2`, o, dicho de otro modo, cuando la expresión Python `s1 in s2` retorna `True`. Escribe una función `mas_largo_comun(s1,s2)`, que retorna el string más largo que, simultáneamente, es substring de s1 y s2. Si hay más de uno puedes retornar cualquiera de esos.
Ejemplos:

- `mas_largo_comun('camion','cancion')` retorna `'ion'`
- `mas_largo_comun('emu','avestruz')` retorna `'e'`
- `mas_largo_comun('puentes','mente')` retorna `'ente'`
- `mas_largo_comun('cae','loro')` retorna `''`
- `mas_largo_comun('puentes','imagenes')` retorna `'en'`

Pregunta 3 (1/3)

Un popular juego criollo es la Rayuela. Se juega entre 2 personas donde cada uno, de manera alternada, lanza un tejo (objeto de metal) hacia una caja llena de barro que está situada a 10 metros de distancia del jugador, donde la caja tiene una lienza al centro. El objetivo es que el tejo quede sobre la lienza.
Voluntariamente harás un programa que permita simular este juego considerando las siguientes reglas:

- Se juega al mejor de 3 sets, es decir, gana el primero en llegar a los 2 sets ganados
- En cada turno del set, los jugadores lanzan el tejo hacia el cajón de barro de manera alternada
  - Si el tejo cae fuera del cuadrado el jugador obtiene 0 puntos
  - Si el tejo cae dentro del cuadrado, pero no sobre la lienza, el jugador obtiene 1 punto
  - Si el tejo cae sobre la lienza el jugador obtiene 2 puntos
  - Si ambos jugadores obtienen el mismo puntaje en el mismo turno, entonces el turno se anula y ninguno obtiene puntaje
  - Gana el set el primer jugador en alcanzar 12 o más puntos al final del turno, con una diferencia de 2 o más puntos sobre su rival

El programa debe pedir el nombre a ambos jugadores. Luego, de manera aleatoria se simula el lanzamiento del jugador 1 y el jugador 2 (0:fuera del cajón, 1:dentro, o 2:sobre la lienza) hasta que uno de los 2 gane la partida. Al final de cada turno, se debe mostrar el marcador del set. Cuando un jugador gana el set, se debe imprimir un mensaje indicando quién ganó y mostrar el marcador de la partida (cuántos sets ha ganado cada uno). Cuando un jugador gana la partida, se debe imprimir un mensaje indicando quién ganó.

```
jugador 1? Miguel
jugador 2? Mauricio
El marcador del set va: Miguel 0 - Mauricio 2
El marcador del set va: Miguel 0 - Mauricio 4
El marcador del set va: Miguel 2 - Mauricio 4
El marcador del set va: Miguel 5 - Mauricio 4
El marcador del set va: Miguel 6 - Mauricio 4
El marcador del set va: Miguel 8 - Mauricio 5
El marcador del set va: Miguel 9 - Mauricio 5
El marcador del set va: Miguel 9 - Mauricio 5
El marcador del set va: Miguel 11 - Mauricio 6
El marcador del set va: Miguel 13 - Mauricio 6
----------------------------------------
El ganador del set es  Miguel
El marcador de la partida va: Miguel 1 - Mauricio 0
----------------------------------------
```

```
El marcador del set va: Miguel 0 - Mauricio 0
El marcador del set va: Miguel 2 - Mauricio 1
El marcador del set va: Miguel 3 - Mauricio 3
El marcador del set va: Miguel 5 - Mauricio 3
El marcador del set va: Miguel 8 - Mauricio 5
El marcador del set va: Miguel 9 - Mauricio 7
El marcador del set va: Miguel 10 - Mauricio 9
El marcador del set va: Miguel 10 - Mauricio 10
El marcador del set va: Miguel 12 - Mauricio 11
El marcador del set va: Miguel 14 - Mauricio 11
-----------------------------------------
El ganador del set es  Miguel
El marcador de la partida va: Miguel 2 - Mauricio 0
-----------------------------------------
El ganador de la partida es Miguel
```