Programación – Evaluación Formativa Diagnóstica – Uso de Listas

A. Enfermer@s

En un hospital, la información del personal de enfermería se almacena en una lista de tuplas, donde cada tupla tiene un identificador único para cada profesional, su nombre y su tipo o especialidad. El tipo puede ser Maternidad (`'M'`), Emergencias (`'E'`) o General (`'G'`).

Por otra parte, los turnos se encuentran almacenados en una lista de tuplas, donde el primer elemento de cada tupla es el identificador del o la profesional, y el segundo es una lista con los números de los bloques en donde tiene turno. Los bloques van del 1 al 10.

```
profesionales = [
    (125,'Ann','M'),
    (271,'Beth','E'),
    (443,'John','G'),
    (105,'Ben','M'),
    (85,'Fer','E'),
    . . .
]
```

```
turnos = [
    (443,[5,8,2]),
    (126,[6,2,4,7]),
    (271,[1,9]),
    (85,[2,9]),
    . . .
]
```

1. Implemente la función `turno_emergencias(profesionales, turnos)`, que recibe como parámetro las listas descritas en el enunciado del problema. La función debe retornar una lista con todos los turnos, sin repetición, en los que trabajan profesionales de Emergencias.

   Ejemplo:

```
>>> print(turno_emergencias(profesionales,turnos))
[1, 9, 2]
```

2. Implemente la función `buscar_profesional(profesional, identificador)`, que recibe como parámetro la lista con la información de los profesionales y el identificador de uno o una de ellos. La función debe retornar el nombre del o la profesional individualizado por ese identificador, o un *string* vacío en caso de que el identificador no se encuentre en la lista.

   Ejemplo:

```
>>> print(buscar_profesional(profesionales, 443))
John
```

3. Implemente la función `jefe(profesionales, turnos)`, que recibe las listas descritas en el enunciado del problema. La función debe retornar el nombre del o la profesional que menor cantidad de turnos tiene asignados. Si hay empate, debe retornar cualquiera de los que empatan en el mínimo.

   Ejemplo:

```
>>> print(jefe(profesionales, turnos))
Beth
```

Programación – Evaluación Formativa Diagnóstica – Uso de Listas

B. Trenes

En la ciudad de Pythonia todos los trenes realizan el mismo recorrido, pasando por cada una de las estaciones cuyo orden se encuentra en la lista `estaciones`.
El tiempo entre una estación y otra depende del tipo de tren. Los trenes *Expreso* tardan 2 minutos en ir desde una estación a la siguiente, los trenes *Normal* demoran 5 minutos, y los trenes de tipo *Corto* tardan 7 minutos. Suponga que el tiempo que el tren se detiene en las estaciones es despreciable, por lo que no será considerado para la solución de este problema.
Los tipos de trenes se encuentran en la lista de tuplas `tipos`. Cada tupla contiene dos elementos: El primero corresponde a un *string* con el tipo de tren, y el segundo a una lista de *strings* con los identificadores únicos de los trenes que pertenecen a ese tipo.
Finalmente, la lista `trenes` contiene tuplas de tres elementos. El primero corresponde al identificador único del tren y los dos siguientes son *strings* que corresponden a las horas y minutos del horario de salida del tren desde la primera estación de la línea: la Estación Central de Pythonia.

```
estaciones = [
   'Estacion Central' ,
   'Sal y Santo' ,
   'Pytronato' ,
   'La Pysterna' ,
   'Pynstein' ,
   'Estadio Nacional' ,
   'Pio-Pio' ,
   'Plaza Python Alto',
   . . .
]
```

```
tipos = [
   ('Expreso', ['E07','F07','F08', ...]),
   ('Normal', ['G00','H00', ...]),
   ('Corto', ['B00','C01', ...])
]
```

```
trenes = [
   ('E07','09','03'),
   ('F07','10','14'),
   ('H00','09','23') ,
   ('B00','08','00') ,
   ('G00','08','46') ,
   ('C01','13','59') ,
   ('F08','11','02'),
   . . .
]
```

Desarrolle la función `proximos_trenes(estaciones, tipos, trenes, est, hor)` donde `est` es un *string* con el nombre de la estación en la que se encuentra el pasajero y `hor` es una tupla en formato `('HH','MM')` que indica la hora actual. La función debe retornar una lista de tuplas con los trenes que aún no han pasado por la estación actual `est` a la hora actual `hor`. El primer elemento de cada tupla corresponde al identificador del tren, y el segundo a los minutos que faltan para el arribo del tren a la estación actual `est`. En caso de no haber trenes que queden por pasar en la hora consultada, se debe retornar una lista vacía.

Ejemplos:

```
>>> proximos_trenes(estaciones, tipos, trenes, 'Pytronato', ('09','06'))
[('E07',1), ('F07',72), ('H00',27), ('F08',120), ('C01',307)]
>>> print proximos_trenes(estaciones, tipos, trenes, 'Pio-Pio', ('11','00'))
[('C01',221), ('F08',14)]
>>> proximos_trenes(estaciones, tipos, trenes, 'Sal y Santo', ('15','00'))
[]
```