Programación—Certamen 2 (CC)- Jueves 12 de Junio de 2014

Nombre: _____ Rol: _____ – _____

1. [30 %] Realice el ruteo de los siguientes programas e indique qué es lo que imprimen. Cada vez que el valor de una variable cambie, escríbalo en una nueva fila de la tabla. Recuerde que si una variable es de tipo string, debe colocar su valor entre comillas simples ' '. Si una variable almacena una función, coloque el nombre de ésta como valor (sin comillas).

***Importante**: La tabla tiene suficientes filas.*

```
def f(x,y = 1):
    if y % 2 == 0:
        return [2*len(x)] + x
    else:
        return x[2:]

def g(a,b):
    while sum(b) != 7:
        if a in b:
            a = a - 2
            b = f(b)
        else:
            a = a + 1
            b = f(b,2)
    return b

a,b = len(set('tarros')),[3,1,4,5]
print g(a,b)
```

global		f		g	
a	b	x	y	a	b

Impresiones

Indique lo que imprimen los siguientes programas.

```
r,s = (2014,3,12),(2014,1,1)
t = (2014,2,1)
print r > s and s < t
```

```
w = {'uno':[1,3],'dos':[2,4],
    'tres':[3,6]}
print w['uno'] + w['tres']
```

```
def funcion1(a):
    a.reverse()
    return a

x = {1:[1, 0], 0:[0, 1]}
r = funcion1(x[0])[1]
print r
```

```
def funcion2(x):
    if len(x) == 1:
        return x
    else:
        return x[-1] + funcion2(x[:-1])

print funcion2('FTW')
```

Programación—Certamen 2 (CC)- Jueves 12 de Junio de 2014

Nombre: _____ Rol: _____ – _____

2. **[35 %]** El servicio de inteligencia de la UTFSM ha detectado una amenaza inminente a sus instalaciones por parte de un grupo terrorista que busca impedir que la universidad se convierta en el mejor centro educacional del mundo. Dada la gravedad de esta amenaza, se ha solicitado que la división de agentes "IWI-131" analice los datos obtenidos por los infiltrados que el servicio de inteligencia posee en otras universidades. Los datos con los que se trabajará se encuentran en un diccionario llamado `terroristas` (variable global) que tiene por llave el identificador de cada terrorista y, por valor, una lista de tuplas que indica las universidades en las que ha sido visto el terrorista junto con la fecha correspondiente.

```
terroristas = {
2352: [('Stanfox', '2010-05-02'), ('Hardyard', '2010-06-07'), (
    'Yon Jopkins', '2010-05-02')],
1352: [('Stanfox', '2010-05-02'), ('Stanfox', '2011-06-08')],
352: [('Hardyard', '2009-03-03')],
22: [('Yon Jopkins', '2012-11-16')]}
```

Un diccionario llamado `experticias` (variable global) que tiene por llave el identificador de cada terrorista y, por valor, la experticia de dicho terrorista.

```
experticias = { 2352: 'TNT', 1352: 'TNT', 352: 'rayos laser', 22: '
   teletransportacion'}
```

a) Desarrolle la función `terroristas_se_conocen(terrorista1, terrorista2)` que reciba como parámetros los identificadores de dos terroristas y que retorne **`True`** si ambos se conocen o **`False`** si no. Dos terroristas se conocen si ambos han sido vistos en el mismo lugar en la misma fecha.

```
>>> terroristas_se_conocen(2352, 1352)
True
>>> terroristas_se_conocen(2352, 352)
False
```

b) Desarrolle la función `terroristas_que_han_estado_en(universidad)` que reciba como parámetro el nombre de una universidad y que retorne un conjunto conformado por los identificadores de los terroristas que han sido visto en la universidad ingresada como parámetro.

```
>>> terroristas_que_han_estado_en('Stanfox')
set([1352, 2352])
>>> terroristas_que_han_estado_en('Prinxton')
set([])
```

c) Desarrolle la función `terroristas_clave()` que retorne una lista de tuplas con los identificadores de los terroristas claves e informe si cada uno de ellos pertenece (**`True`**) o no (**`False`**) a una "sleeper cell". Se considera que un terrorista es clave si es el único que posee cierta experticia. Se considera que un terrorista pertenece a una "sleeper cell"si es que no conoce a ningún otro terrorista.

```
>>> terroristas_clave()
[(22, True), (352, True)]
```

Programación—Certamen 2 (CC)- Jueves 12 de Junio de 2014

Nombre: _____ Rol: _____ – _

3. [35 %] La gran maratón de Chago city es una de las carreras más importantes a nivel mundial. Debido a la gran cantidad de competidores que reúne este evento, se han generado las siguientes estructuras para ayudar con la organización de éste:

- Diccionario con los inscritos, donde se almacena el número del corredor y los datos de éste.

```
inscritos = { #num_corredor: (rut,nombre,apellido,id_categoria,edad)
                1001: ('1111111-2', 'Carlos', 'Caszely', 2, 55),
                1002: ('223244-4', 'Marcelo', 'Rios', 3, 45),
                2129: ('3838292-1', 'Ivan', 'Zamorano', 4, 38),
                4738: ('5940301-2', 'Erika', 'Olivera', 5, 48),
                8883: ('3843993-1', 'Condor', 'ito', 3, 22),
                231:  ('9492922-2', 'Pepe', 'Antartico', 3, 30)
}
```

- Diccionario con las categorías, el cual almacena la distancia que se corre y el premio para el ganador.

```
categorias = { #  id_categoria: (distancia, premio)
                1: ('1k', 10000),
                2: ('5k', 20000),
                3: ('10k', 450000),
                4: ('21k', 100000),
                5: ('42k', 250000)
}
```

- Lista de resultados, donde se registra el número del corredor y el tiempo que logró.

```
#           [ (num_corredor, tiempo) ]
resultados =  [(1001, '00:30:12'), (1002, '00:55:43'), (2129,
    '01:45:23'), (4738, '03:05:09'), (8883, '00:31:33'), (231,
    '00:39:45')]
```

a) Desarrolle la función `competidores_edad(inscritos, categorias, min_edad, max_edad)` que reciba el diccionario `inscritos`, el diccionario `categorias` y los valores enteros `min_edad` y `max_edad` (que representan la máxima y mínima edad). La función debe retornar una lista de tuplas de todos los competidores que se encuentren entre la edad mínima y máxima (incluyéndolos), donde cada tupla contenga el nombre, apellido y la distancia a correr de un individuo.

```
>>> competidores_edad(inscritos,categorias,25,40)
[('Pepe', 'Antartico', '10k'), ('Ivan', 'Zamorano', '21k')]
```

b) Desarrolle la función `tiempo_competidor(inscritos, resultados, rut)` que reciba el diccionario `inscritos`, la lista de tuplas `resultados` y el string `rut`. La función debe retornar el tiempo, como cadena de texto, de un competidor en particular.

```
>>> tiempo_competidor(inscritos,resultados,'9492922-2')
'00:39:45'
```

c) Desarrolle la función `ganador_categoria(inscritos, categorias, resultados, distancia)` que reciba el diccionario `inscritos`, el diccionario `categorias`, la lista de tuplas `resultados` y el string `distancia`. La función debe retornar una tupla con el ganador de cierta categoría, indicando el nombre, apellido y premio obtenido.

```
>>> ganador_categoria(inscritos, categorias, resultados, '10k')
('Condor', 'ito', 450000)
```