Programación—Certamen 3 (CC) - Viernes 8 de Enero de 2016

Nombre: _____ Rol: _____ – _

1. **[20 %]** El banco Pythonbank maneja información de sus clientes en el archivo `clientes.txt`, con una estructura del tipo `rut#nombre#direccion`. Además cuenta con el archivo `movimientos.txt`, con una estructura del tipo `rut,producto,movimiento,aaaa-mm-dd`, donde se almacenan los cargos (movimientos negativos) y abonos (movimientos positivos) que se realizan en los productos de los clientes. A partir de estos archivos se desea generar un nuevo archivo `productos.txt`, con una estructura del tipo `rut/nombre/cantproductos/saldo`, donde se resumirá la información presente en los archivos antes mencionados.

A continuación se presentan las líneas de código que resuelven este problema, pero están desordenadas. Usted debe ordenarlas e indentarlas (dejar los espacios correspondientes de python) para que ambas funciones estén correctas.

La función `prod_cli`, a partir del archivo de movimientos, retorna una tupla con la cantidad de productos y el saldo final asociado a un rut dado. La función `archivo_prod` crea el archivo `productos.txt`, recibiendo como parámetro el nombre de los archivos antes descritos.

```
if p not in c:
mov_banco.close()
sumaprod += 1
saldo = saldo + int(s)
r,p,s,_ = li.strip().split(',')
c = set()
saldo = 0
def prod_cli(rut, movimiento):
c.add(p)
if r == rut:
return sumaprod, saldo
sumaprod = 0
for li in mov_banco:
mov_banco = open(movimiento)
```

```
cli_banco.close()
r, n, _ = li
cli_banco = open(cli)
prod.write(f.format(r, n, cant, s))
cant, s = prod_cli(r, movimiento)
for li in cli_banco:
li = li.strip().split('#')
prod.close()
prod = open('productos.txt', 'w')
f = '{0}/{1}/{2}/{3}\n'
def archivo_prod(cli, movimiento):
```

Programación—Certamen 3 (CC) - Viernes 8 de Enero de 2016

Nombre: _____ Rol: _____ – _

2. **[40 %]** Haciendo la tarea 3 de programación (WhatsAppy), los estudiantes se dieron cuenta del problema que les provocaba los distintos formatos al guardar mensajes en Whatsapp. Por este motivo se decidió crear una nueva aplicación de mensajería, donde los mensajes (no acciones) de un chat son almacenados en distintos archivos (uno por fecha) en el formato `hh:mm usuario: mensaje`. Por ejemplo, de las fechas 2015-12-31 y 2016-01-01 se obtuvo los siguientes archivos:

`2015-12-31.txt`

```
11:05 Cesar: hola alguien sabe algo de Miguel
11:27 Hector: no se pero seguramente trabaja en su tesis
13:10 Cesar: es que todavia no corrige :/
16:10 Miguel: NO SE PREOCUPEN pasare toda la noche corrigiendo
```

`2016-01-01.txt`

```
00:05 Miguel: Ahora si feliz anho a todos
```

Ahora se ha decido crear WhatsAppy v2.0, donde usted debe programar las siguientes funcionalidades:

a) Escriba la función `fecha_a_nomarch(fecha)` que reciba como parámetro una tupla con una `fecha` y retorne el nombre del archivo en la fecha indicada con el formato *`'AAAA-MM-DD.txt'`*.

```
>>> fecha_a_nomarch((2015, 12, 31))
'2015-12-31.txt'
```

```
>>> fecha_a_nomarch((2016, 1, 1))
'2016-01-01.txt'
```

b) Escriba la función `rango(fecha, usuario)` que reciba como parámetro una tupla de `fecha` y el nombre de un `usuario`. La función debe retornar una tupla correspondiente al rango horario en el cual el usuario intervino en el chat. Sólo indicar el valor de la hora, como número **entero**, de los mensajes borde. Si el usuario no ha realizado intervenciones en la fecha indicada, retornar una tupla `(24, 24)`.

```
>>> rango((2015, 12, 31), 'Cesar')
(11, 13)
```

```
>>> rango((2015, 12, 31), 'Hector')
(11, 11)
```

c) Escriba la función `top_por_mensaje(fechas, N)` que reciba como parámetro una lista de tuplas de `fechas` y un número entero `N`. La función debe retornar una lista de tuplas con los `N` usuarios que más mensajes escribieron entre todas las `fechas` indicadas, ordenados de mayor a menor número de mensajes. Cada tupla está compuesta por el número de mensajes y el usuario. Si dos o más usuarios empatan en el último puesto, incluya cualquiera de los empatados.

Nota: asuma que todos los archivos, de las fechas indicadas, existen.

```
>>> top_por_mensaje([(2016, 1, 10), (2015, 12, 31)], 2)
[(2, 'Cesar'), (1, 'Miguel')]
>>> top_por_mensaje([(2016, 1, 1), (2015, 12, 31),  (2016, 1, 5)], 4)
[(2, 'Cesar'), (2, 'Miguel'), (1, 'Hector')]
```

Hint: recuerde que `lista.sort()` ordena la `lista` de menor a mayor.

Programación—Certamen 3 (CC) - Viernes 8 de Enero de 2016

Nombre: _____ Rol: _____ – _____

3. [**40 %**] Complementando la pregunta 2, se le solicita:

a) Escriba la función `agregar_mensaje(msm)` que reciba como parámetro una lista con el mensaje de un usuario. La función debe agregar el mensaje del usuario al final del archivo de la `fecha` correspondiente. La función no tiene retorno.

```
>>> #fecha (aaaa, mm, dd), 'hh:mm', 'usuario', 'mensaje'
>>> msm = [(2015, 12, 31), '23:55', 'Miguel', 'termine de corregir :D']
>>> agregar_mensaje(msm)
>>>
```

b) Escriba la función `chat_censurar2(fecha, palabra)` que reciba como parámetros una tupla de `fecha` y una `palabra`. La función debe censurar toda aparición de la `palabra` en los **mensajes** del `archivo`, reemplazándola cada letra de la palabra por una `@`. Censure las palabras independiente si están en mayúsculas o minúscula. La función no tiene retorno.

Nota: asuma que no hay signos de puntuación.

```
>>> chat_censurar2((2015, 12, 31), 'se')
>>>
```

2015-12-31.txt

```
11:05 Cesar: hola alguien sabe algo de Miguel
11:27 Hector: no @@ pero seguramente trabaja en su tesis
13:10 Cesar: es que todavia no corrige :/
16:10 Miguel: NO @@ PREOCUPEN pasare toda la noche corrigiendo
23:55 Miguel: termine de corregir :D
```

c) Escriba la función `intervenciones_usuario(fechas, usuario)` que reciba como parámetro una lista de tuplas de `fechas` y el nombre de un `usuario`. La función debe crear un archivo `chat_[usuario].txt` donde `[usuario]` es el nombre del usuario recibido como parámetro. El contenido de este archivo deben ser todos los mensajes de chat del `usuario` en las distintas `fechas` ordenados desde el más antiguo al más actual. La función no tiene retorno.

```
>>> intervenciones_usuario([(2016, 1, 1), (2015, 12, 31)], 'Miguel')
>>>
```

chat_Miguel.txt

```
16:10 Miguel: NO @@ PREOCUPEN pasare toda la noche corrigiendo
23:55 Miguel: termine de corregir :D
00:05 Miguel: Ahora si feliz anho a todos
```