Programación—Certamen 3 CS - Miércoles 23 de Diciembre de 2015

Nombre: Rol: –

1. **[25 %]** El diario popular de pytonia, **La puerta**, desea escuchar la voz del pueblo y premiar a los programas y personajes más queridos y de buen desempeño durante el año 2015, a través del concurso `la python de oro`.

Para ello dispone de un archivo de texto, `nominados.txt` donde cada línea del archivo tiene la siguiente estructura: categoría del premio#programa nominado#votos obtenidos. Ver ejemplo del archivo más abajo.

nominados.txt

```
mejor programa#Y tu que programarias#234
mejor serie o teleserie#Pytuca sin tuplas#673
mejor programa#En su propia lista#341
mejor programa#Archivo especial#567
mejor serie o teleserie#python a la deriva#973
mejor serie o teleserie#la tuplara#473
...
```

A partir de un archivo, como el descrito anteriormente, se desea generar el archivo `ganadores.txt`, el cual debe poseer en cada línea el nombre de la categoría y el ganador de dicha categoría, como se muestra a continuación:

`categoria-ganador`

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

Tener en cuenta que la función *crear_file* recibe como parámetros el nombre del archivo que contiene los nominados y una lista con las categorias a premiar. Esta función tiene por objetivo generar un diccionario con la información obtenida desde el archivo, cuya clave es la categoria y el valor una tupla con el nombre del nominado y los votos obtenidos. La función no retorna nada, sólo crea el archivo `ganadores.txt`

La función *get_win* recibe 2 parámetros, un string con el nombre de la categoría de la cual se desea obtener el ganador y un diccionario como el generado en la función anterior. La función retorna el nombre del ganador de la categoría recibida como parámetro.

crear_file

```
resu.close()
def crear_file(file, categorias):
resu = open("ganadores.txt","w")
if l[0] not in d:
for cate in categorias:
d[l[0]]=[]
d = {}
arch.close()
for linea in arch:
l = linea.strip().split("#")
s = '{0}-{1}\n'
arch = open(file)
resu.write(s.format(cate,win))
x = tuple(l[1:])
win = get_win(cate, d)
d[l[0]].append(x)
```

get_win

```
mayor = votos
win = x[0]
def get_win(categoria, diccio):
return win
win = ""
votos = int(x[1])
mayor = -1
if votos>mayor:
lista = diccio[categoria]
for x in lista:
```

Programación—Certamen 3 CS - Miércoles 23 de Diciembre de 2015

Nombre: _____ Rol: _____ – _

2. **[30 %]** El ejército del imperio de Pythonia se encuentra en guerra contra el imperio de Rubyonia. De esta manera, se ha dedicado a capturar información del enemigo para adelantarse a posibles emboscadas. Algunos mensajes interceptados se encuentran en el archivo `mensajes.txt`. Cada línea se compone del mensaje cifrado, y la IP de origen del mensaje.

Cada mensaje ha sido codificado usando el algoritmo de **Cifrado César** el cual permite cifrar un mensaje a través de la sustitución de cada letra por otra ubicada cierto número de posiciones más adelante, (considere el abecedario cíclico, es decir que después de la Z viene la A y así sucesivamente). Por ejemplo, para el mensaje **Hola Mundo**, si se considera un desplazamiento de 3, cada letra debe ser reemplazada por la letra situada 3 posiciones más adelante en el abecedario, por lo tanto, se codificaría de la siguiente manera: Mensaje original = "hola mundo" =¿mensaje codificado = "krod pxqgr"

Por su parte, el archivo `IPS.txt` contiene en cada línea, la dirección IP de envío del mensaje, el sector al que pertenece esa IP y el desplazamiento para descifrar los mensajes correspondientes a esa IP

Los siguientes archivos son un ejemplo de lo anterior:

`mensajes.txt`

```
dqodctfgct gn egpvtq@@198.43.23.1
jrgtxhfif fq xzw@@143.32.43.55
fsibwcb rs qccfrwboqwcb sb pogs 9@@154.33.33.12
dwdfdu sodcd 1@@123.32.22.10
```

`IPS.txt`

```
123.32.22.10,sector1=3
198.43.23.1,sector1=2
143.32.43.55,sector2=5
154.33.33.12,sector1=14
```

Para ayudar al Imperio Pytonia, se le pide escribir las siguientes funciones:

a) Desarrolle la función `descifrar_mensaje(mensaje_cifrado, desplazamiento)` la cual recibe como parámetro un string con el mensaje cifrado y un entero que indica el desplazamiento que le fue realizado. La función debe retornar el mensaje descifrado. Hint: los mensajes a descifrar no tienen signos de puntuación ni caracteres extraños. Si en el mensaje aparece un dígito o espacio en blanco se deben mantener. Asuma que existe una variable global de tipo string llamado **abecedario** como el que se describe a continuación:

abecedario = 'abcdefghijklmnopqrstuvwxyz'

```
>>> descifrar_mensaje('erpedughdu ho fhqwur gh od fdslwdo', 3)
'bombardear el centro de la capital'
```

b) Desarrolle la función `datos_por_sector(archivo_mensajes, archivo_ips)` la cual recibe el nombre del archivo con los mensajes cifrados y el nombre del archivo con las IPs. La función retorna un diccionario cuya llave es el sector del emisor del mensaje, y el valor una lista de mensajes descifrados para ese sector.

```
>>> datos_por_sector('mensajes.txt', 'ips.txt')
{'sector1': ['atacar plaza 1', 'bombardear el centro', 'reunion de
   coordinacion en base 9']},
 'sector2': ['emboscada al sur'],  ...
}
```

Programación—Certamen 3 CS - Miércoles 23 de Diciembre de 2015

Nombre: _____ Rol: _____ – _

3. **[45 %]** Una prestigiosa clínica ha decidido renovar su forma de agendar horas por lo que ha instaurado el uso de archivos de texto para esta misión. Para ello, se han elaborado los siguientes documentos:

- `medicos.txt` contiene una estructura la cual permite almacenar los datos de un médico en cada una de sus filas con la siguiente estructura nombre;rut;especialidad.
- `agenda.txt` contiene una estructura la cual permite almacenar los datos de la consulta de un paciente para cierto médico en un día-hora, siguiendo la siguiente estructura rutMedico#rutPaciente#fecha@hora.

`medicos.txt`

```
gonzalo medel;5555555-5;Tr
maria soto;9999999-9;Card
manuel lobos;6565666-6;Oft
...
```

`agenda.txt`

```
5555555-5#23232323-2#2015/12/04@12:30
9999999-9#23232323-2#2015/12/11@11:00
5555555-5#1111111-1#2015/12/09@16:30
...
```

En base a los archivos presentados anteriormente, se le solicita:

a) Desarrolle la función `asignacion(agenda)`, que reciba un string con el nombre del archivo que contiene las horas de la consulta. La función debe retornar un diccionario donde la llave corresponda al rut del médico y su valor una lista de tuplas con los pacientes a atender. La tupla por cada paciente a atender debe llevar el rut del paciente en string, la tupla de enteros (aaaa,mm,dd) y un string con la hora de atención.

```
>>> asignacion('agenda.txt')
{'9999999-9': [('23232323-2', (2015, 12, 11), '11:00')], '5555555-5':
   [('23232323-2', (2015, 12, 4), '12:30'), ('1111111-1', (2015, 12, 9)
   , '16:30')]}
```

b) Desarrolle la función `horas_paciente(rut, medicos, agenda)` que reciba como parámetros un string correspondiente al rut del paciente y los nombres de los archivos correspondiente a médicos y agenda. La función debe retornar un conjunto de tuplas con los datos de los médicos con quien tiene hora agendadas. Si el paciente a consultar no tiene horas agendadas, retorna un conjunto vacio.

```
>>> horas_paciente('1111111-1','medicos.txt','agenda.txt')
set([('gonzalo medel', 'Tr', (2015, 12, 9), '16:30')])
>>> horas_paciente('12345678-9','medicos.txt','agenda.txt')
set([])
```

c) Desarrolle la función `agregar_hora(tupla)` que reciba una tupla con la siguiente estructura: el rut del médico (en string), el rut del paciente (en String), una fecha (en formato tupla de enteros aaaa,mm,dd), hora (en string) y el nombre del archivo agenda (string). La función debe agregar una nueva asignación para médico-paciente en la fecha y hora dada. Si el paciente ya posee una consulta para ese médico (independiente de la hora-día) la función debe retornar False. En caso contrario, debe agregar una nueva línea al archivo agenda y retornar True.

```
>>agregar_hora(('9999999-9','16128912-2',(2015,12,11),'12:00','agenda.
   txt'))
True
```