Programación—Certamen 3 - Jueves 29 de Agosto de 2013

Nombre: _____ Rol: _____ – _____

1. [30 %] Indique qué es lo que imprimen los siguientes programas.

```
fecha = '21-12-2013'
d,m,a = map(str, fecha.split('-'))
print d+m+a
```

```
animales = ['Perro', 'Gato', 'Raton']
c = ' '.join(animales)[6]
print c*2
```

```
a = "anacleta la bicicleta"
b = "\n".join(a.split())
print b[:len(a)]
```

```
p = "esternocleidomastoideo"
k = p.replace("e", "a").replace("a", "u")
print k[::3]
```

Preguntas de archivos

1. Considere el archivo `permisos.txt`. Indique en el cuadro de más abajo qué es lo que se muestra al ejecutar el siguiente segmento de código:

```
arch = open('permisos.txt')
cont = 0
for f in arch:
    cont += 1
    data = f.strip().split(':')
    x, y, z = data[1].split('-')
    m = 'auto{0}: {1} valor: ${2}'
    print m.format(cont, x, z)
arch.close()
```

permisos.txt

```
A325181:MAZDA6-6millones-201591
A340124:LANCER-6millones-123591
A330119:B180-10millones-256379
A023072:A3-9millones-213591
```

2. Indique en el cuadro de más abajo cómo queda el archivo `info.txt` después de ejecutar el siguiente segmento de código:

```
arch = open("datos.txt")
info = open("info.txt", "w")
for l in arch:
        v = l.strip().split(":")
        t = map(float, v)
        k = ".".join(map(str,t))
        if sum(t) < len(l)*4:
                info.write(k +"\n")
arch.close()
info.close()
```

datos.txt

```
12:34:76:2
10:2:15
356:7:7:06:4
1:1:3:5
```

Programación—Certamen 3 - Jueves 29 de Agosto de 2013

Nombre: _____ Rol: _____ – _

2. **[35 %]** Una prestigiosa universidad mantiene una lista de alumnos en un archivo `alumnos.txt`, donde se almacenan el ROL y el nombre de los alumnos en el formato `rol:nombre`. Además, por cada alumno, hay un archivo `rol.txt`, donde el nombre es el ROL del alumno. Este último tiene los ramos tomados por el alumno y las notas obtenidas en el formato `ramo,nota1 nota2 ...`, como muestra el siguientes **ejemplo**:

`alumnos.txt`

```
201308054:Juan Duarte
201308012:Loreto Godoy
201308021:Alvaro Luzzi
201308002:Marcelo Clavel
```

`201308054.txt`

```
IWI-131,56 85 97
MAT-010,89 21 23 54
FIS-130,10 90
ILI-110,70 80 90 68 71
```

En el ejemplo, además existen los archivos `201308054.txt`, `201308012.txt`, `201308021.txt` y `201308002.txt` con sus respectivos ramos y notas.

a) Desarrolle una función `notas(alumnos, nombre)` que recibe el nombre del archivo de `alumnos` y el `nombre` de un alumno. La función debe retornar un diccionario donde las llaves son las siglas de los ramos tomados y los valores listas con las notas obtenidas del alumno especificado.

```
>>> notas("alumnos.txt", "Juan Duarte")
{'ILI-110': [70, 80, 90, 68, 71], 'IWI-131': [56, 85, 97],
'MAT-010': [89, 21, 23, 54], 'FIS-130': [10, 90]}
```

b) Desarrolle una función `mejor_alumno(alumnos, ramo)` que recibe el nombre del archivo de `alumnos` y la sigla de un ramo especificado. La función debe retornar el nombre del alumno que mejor promedio obtuvo en dicho ramo. Si el ramo no ha sido tomado por algún alumno, considerar nota 0 en ese caso.

```
>>> mejor_alumno("alumnos.txt", "ILI-110")
"Juan Duarte"
```

c) Desarrolle una función `agregar_ramo(rol, ramo, notas)` que recibe el ROL de un alumno, la sigla de un ramo y una lista con las notas de dicho ramo. La función debe agregar dicho ramo y sus notas al archivo del alumno de ROL `rol`. La función no retorna nada.

```
>>> agregar_ramo("201308054",
    "DEW-010", [100, 54, 79])
>>>
```

`201308054.txt`

```
IWI-131,56 85 97
MAT-010,89 21 23 54
FIS-130,10 90
ILI-110,70 80 90 68 71
DEW-010,100 54 79
```

Programación—Certamen 3 - Jueves 29 de Agosto de 2013

Nombre: _____ Rol: _____ – _

3. [35 %] Hace cientos de años, los humanos fueron casi masacrados por Titanes, seres no inteligentes que devoran humanos. La sociedad se ha guarecido en ciudades protegidas por grandes murallas y donde la única forma de contraatacar es a través del escuadrón de reconocimiento. Para mantener un orden, este escuadrón mantiene un registro de todos sus miembros en un archivo con la estructura `nombre:edad:genero:ranking,rango,habilidad`, como en el siguiente archivo de ejemplo.

escuadron.txt

```
Eren Yeager:15:m:5,soldado raso,constancia
Mikasa Ackerman:15:f:1,soldado raso,equilibrio
Levi:30:m:1,capitan,liderazgo
Reiss:15:f:10,soldado raso,curacion
Petra Ral:28:f:4,subcomandante,vision de campo
```

Tras salir a combatir existen muchas bajas, por eso necesitan mantener un registro actualizado constantemente. Para ello:

a) Desarrolle una función `listar_bajas(rango, registro, bajas)`, la cual recibe el `rango` de los soldados a listar, un string `registro` que especifica el nombre del archivo con los soldados y una lista de **`True`** y **`False`**, siendo **`True`** si el soldado aún vive o **`False`** si no. La función retorna una lista con tuplas con el `nombre`, `edad`, `genero` y `ranking` de los soldados muertos (bajas) en el `rango` especificado.

```
>>> listar_bajas("soldado raso", "escuadron.txt", [False,True,False])
[('Eren Yeager', '15', 'm', '5'), ('Reiss', '15', 'f', '10')]
```

b) Desarrolle una función `promedio_edad(bajas)`, la cual recibe la lista `bajas` (como la generada en la pregunta *a)*) y retorna el promedio de edad de los soldados en dicha lista. Redondee el promedio.

```
>>> bajas = listar_bajas("soldado raso", "escuadron.txt", [False,True,False])
>>> promedio_edad(bajas)
15.0
```

c) Cree la función `actualizar_registro(registro, bajas)`, la cual recibe como parámetro un string correspondiente al nombre del archivo con los datos de las tropas (`registro`) y una lista de tuplas con los datos de los soldados de bajas (`bajas`). Esta función debe crear un nuevo archivo, llamado `nuevo_ + nombre_del_archivo`, ver ejemplo. Este archivo debe contener sólo los soldados sobrevivientes. La función no tiene retorno.

```
>>> actualizar_registro("escuadron.txt", bajas)
>>>
```

nuevo_escuadron.txt

```
Mikasa Ackerman:15:f:1,soldado raso,equilibrio
Levi:30:m:1,capitan,liderazgo
Petra Ral:28:f:4,subcomandante,vision de campo
```