Programación—Certamen 3 (CC-CSJ), jueves 22 de noviembre de 2012

Nombre: _____ Rol: _____ – _

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

```
p = 'cara'
p.replace('ca', 'pe')
p.replace('r', 's')
print p[::2].upper()
```

```
linea = '1:Juanito:100-29-88'
filtro = linea.split(':')
s = map(int, filtro[2].split('-'))
print sum(s)/3
```

```
elem = ['Juan','Pedro','Luis']
c=' '.join(elem)[5].replace('P','C')
print c
```

```
b = 'Le mer estebe serene'
b.replace('e','i').replace('i','a')
print b
```

Preguntas de archivos

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

```
arch = open('partes.txt')
cont = 0
for m in arch:
  cont += 1
  datos = m.strip().split(':')
  x, y = datos[1].split('-')
  p = datos[2][:-1]
  k = '{0}-{1} paga {2}'
  print k.format(cont, x, p)
arch.close()
```

partes.txt

```
jperez:SJHD43-2:143000P
jvivar:CRZT65-4:59000N
fdetal:QWER54-1:120000N
lsoto:PYTH44-6:39990P
```

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

```
arch = open('origen.txt')
arch2 = open('destino.txt', 'w')
f = '{0} {1}->{2}\n'
for k in arch:
  datos = k.strip().split(';')
  a = datos[2].split('-')
  z = int(round(sum(map(int, a))/3.0))
  b = datos[1].split()
  arch2.write(f.format(b[1],b[0],z))
arch.close()
arch2.close()
```

origen.txt

```
201073087-6;Juan Perez;68-32-11
201185077-5;Pedro Soto;98-65-99
2965099-2;Luis Saez;32-55-56
```

Programación—Certamen 3 (CC-CSJ), jueves 22 de noviembre de 2012

Nombre: _____ Rol: _____ – _

2. **[35 %]** El profesor *Salomon Troncoso* ha decidido medir la asistencia de cada uno de sus estudiantes en las últimas 10 sesiones de la asignatura.

El registro de la asistencia se mantiene en una lista de tuplas, donde cada tupla consiste en 2 valores: el `RUT` del estudiante y el nombre con la asistencia. El formato del nombre y la asistencia, tiene la siguiente estructura: `nombres apellido-asistencias`

```
asistencias = [
        ('18647202-3', 'Sr.Pedro Salinas-1:1:0:1:1:1:0:1:1:0'),
        ('18905029-9', 'Srta.Paola Cecilia Rosas-1:0:1:0:1:1:0:1:1:0'),
        ('18293884-1', 'Sr.Diego Moya-1:1:1:1:1:1:1:1:1:1'),
        ('18293884-1', 'Srta.Carolina Zamora-1:1:1:1:0:1:1:0:1:0'),
        ('18293884-1', 'Sr.Pedro Diego Morales-1:1:1:1:1:1:1:1:1:1'),
         ]
```

Note que hay estudiantes con uno y dos nombres (como es el caso de *Paola Cecilia* y *Pedro Diego*). La asistencia es una secuencia de unos y ceros separados por dos puntos (`:`) que indica si el estudiante asistió (`1`) o no asistió (`0`) a cada una de las 10 sesiones. Por ejemplo el estudiante *Pedro Salinas* asistió la sesión 1, 2, 4, 5, 6, 8, 9 y se ausentó la sesión 3, 7 y 10.

Considere que la lista `asistencias` puede tener muchos registros, no sólo los del ejemplo.

a) Escriba la función `asistencias_sesiones(asistencias, rango)`. El parámetro `rango` es una tupla de 2 elementos `(inicio, fin)` y denota un rango de sesiones. Por ejemplo, si `rango` es `(2, 5)` indica que se debe considerar desde la sesión 2 a la 5 (la 5 incluida). Si el rango es `(4,4)` refiere sólo a la sesión 4. Si el rango es `(5, 2)` es inconsistente, debe mostrar el mensaje `'Incorrecto'`. La función debe retornar una lista con la cantidad de asistencias de cada estudiante en ese rango de sesiones.

```
>>> asistencias_sesiones(asistencias, (2, 5))
[3, 2, 4, 3, 4]
```

Del ejemplo anterior, entre las sesiones 2 y 5: el estudiante 1 asistió a 3 sesiones, el estudiante 2 a 2 sesiones, etc.

```
>>> asistencias_sesiones(asistencias, (4, 4))
[1, 0, 1, 1, 1]
>>> asistencias_sesiones(asistencias, (5, 2))
'Incorrecto'
```

b) Escriba la función `estudiantes_responsables(asistencias)` que retorne una lista con los estudiantes que fueron a las 10 sesiones. Cada elemento de la lista debe tener la siguiente estructura: `apellido, primer_nombre`

```
>>> estudiantes_responsables(asistencias)
['Moya, Diego', 'Morales, Pedro']
```

c) Escriba la función `generar_cartas(asistencias)` que genere un archivo por cada estudiante con la cantidad de asistencias, ausencias y el porcentaje asociado. El nombre de cada archivo debe ser `RUT.txt`, donde RUT es el rut del estudiante. Por ejemplo para la carta de *Paola Cecilia Rosas* se debería generar un archivo llamado `18905029-9.txt` con el siguiente formato y contenido.

```
Srta.Paola Cecilia, tiene 6 asistencias y 4 ausencias.
60.0 % de asistencia.
```

Note que el o los nombres van con Sr. o Srta., sin el apellido. La función no debe retornar nada.

Programación—Certamen 3 (CC-CSJ), jueves 22 de noviembre de 2012

Nombre: _____ Rol: _____ – _

3. [35 %] En las votaciones del año 2012 se enfrentan 2 candidatos: el *candidato A* y el *candidato B*. La votación se realiza en varias zonas de Chile. Por cada una de estas zonas se ha generado un archivo con los votos emitidos llamado `zonaX.txt` (donde `X` es el número de la zona). Cada línea del archivo posee la siguiente estructura:

`candidato mesa/hora_emision_voto:minuto_emision_voto`

`zona1.txt`

```
A 3/12:09
B 4/13:25
A 2/09:40
```

`zona2.txt`

```
B 2/11:00
A 3/15:33
B 2/16:55
```

`zona3.txt`

```
A 3/18:14
B 3/16:22
A 2/10:26
```

(Considere que cada archivo puede tener muchos votos, no sólo los del ejemplo).

Por otro lado, el nombre de los archivos de cada zona se encuentra en la lista `lista_archivos`.

```
lista_archivos = ['zona1.txt', 'zona2.txt', 'zona3.txt']
```

Por supuesto, `lista_archivos` puede tener muchas zonas de Chile, no sólo las del ejemplo.

a) Escriba la función `juntar_archivos(lista_archivos)` que permita generar un archivo llamado `votaciontotal.txt` con todos los votos de todas las zonas de Chile. La estructura de este nuevo archivo debe ser: `zona:candidato/mesa/hora_emision_voto`

La `zona` se puede obtener por medio del nombre del archivo.

Ejemplo del archivo `votaciontotal.txt`

```
zona1:A/3/12
zona1:B/4/13
zona1:A/2/09
zona2:B/2/11
...
```

Esta función no debe retornar nada, sólo debe realizar lo indicado anteriormente.

b) Utilizando el archivo `votaciontotal.txt` (creado anteriormente), escriba la función `contar_votos(nombre_archivo)`, que retorne una tupla de 3 elementos: los votos totales del candidato `A`, los votos totales del candidato `B` y el ganador.

```
>>> contar_votos('votaciontotal.txt')
(5, 4, 'A')
```

c) Considere los siguientes periodos:

- Periodo 1: de 8:00 a 14:59 hrs.
- Periodo 2: de 15:00 a 18.59 hrs.

Escriba la función `periodo(nombre_archivo)` que retorne el número del periodo en el cual se registró la mayor votación del candidato ganador.

```
>>> periodo('votaciontotal.txt')
1
```