## Programación—Certamen Rec.(CC-CSJ), jueves 29 de noviembre 2012

Nombre: Rol: –

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

```
a = 2
b = [1, 10, 2.3]
print b[a] * a
```

```
a = 21 / 2.0
c = len(str(a)) % 3
print not c == 1
```

```
p = 'envio;la;carta;maria'
p = p.split('a;')
print 'a-'.join(p[-2:])
```

```
s = 'paralelepipedo'
p = s.replace('r', '').split('l')
print p[1]
```

```
d = {0: '123', 2: '456',
     4: '789', 6: '190'}
print d[int(d[int(d[2][2])][-1])]
```

```
p = (1,1,2012)
q = (1,12,2012)
print p < q
```

```
a = set()
dias = {
    "Enero": (2,3,4,5,8,9,6),
    "Febrero": (2,3,4,5,6)
}
for d in  dias:
    a.add(dias[d][-1])
print a
```

```
def d(n, t):
    return t(n)

def k(n):
    fa = 1
    for i in range(1,n+1):
        fa *= i
    return fa

print d(3, k)
```

## Programación—Certamen Rec.(CC-CSJ), jueves 29 de noviembre 2012

Nombre: _____ Rol: _____ – _

**2.** **[35 %]** Una forma segura de enviar datos a través de internet es mediante la encriptación de datos. Para ello existen funciones que permiten codificar y decodificar.

Una forma de realizar esto es asociar un único valor a un caracter y luego sumar un valor *aleatorio* para que no sea fácil de adivinar.

Lo que usted debe hacer es implementar las funciones para codificar y decodificar un mensaje considerando lo siguiente:

- El diccionario `caracteres` contiene los valores enteros asociados a cada caracter. La llave del diccionario es un carácter, y el valor es el número entero asociado. Note el último carácter del diccionario.

  ```
  caracteres = {'a': 0, 'b': 1, 'c': 2, ..., 'z': 25, '_': 26}
  ```

- El número aleatorio que se debe sumar se encuentra en una lista `aleatorios`. Al primer caracter del mensaje se le debe sumar el valor en la primera posición de la lista (`aleatorios[0]`) y así consecutivamente. Si se acaban los números aleatorios, es decir, si el mensaje es más largo que la lista de números aleatorios, se debe volver a recorrer la lista desde el comienzo.

  ```
  aleatorios = [0, 1, 2, 3, 4, 5, 6, 7]
  ```

Asuma que todos los caracteres del mensaje original están en el diccionario `caracteres`. Además, considere que la lista `aleatorios` puede tener muchos valores, no sólo los del ejemplo.

*a*) Escriba la función `codificar(mensaje, caracteres, aleatorios)`, que retorne una lista de números correspondientes al mensaje codificado.

```
>>> codificar('hola_que_tal', caracteres, aleatorios)
[7, 15, 13, 3, 30, 21, 26, 11, 26, 20, 2, 14]
```

*b*) Escriba la función `decodificar(mensaje_codificado, caracteres, aleatorios)`, que retorne el mensaje original.

```
>>> mensaje_codificado = codificar('hola_que_tal', caracteres, aleatorios)
>>> decodificar(mensaje_codificado,caracteres,aleatorios)
'hola_que_tal'
```

## Programación—Certamen Rec.(CC-CSJ), jueves 29 de noviembre 2012

Nombre: _____ Rol: _____–_

**3.** [35 %] Los subtítulos de una película son archivos de texto plano que tiene un formato especial para que cualquier reproductor multimedia pueda mostrarlos durante la reproducción de una película. El nombre de este tipo de archivos tiene el formato `nombre_archivo.srt`.

El formato que tiene este tipo de archivo es: identificación del subtítulo (como `Sx`, donde `x` es el número del subtítulo), tiempo de inicio y fin (expresado en `hora:minutos:segundos,milisegundos`, una línea en blanco y luego se repite el formato para cada subtítulo. Observe el archivo de ejemplo `los_simpsons.srt` que está a derecha.
Los subtítulos **no deberían** solaparse unos con otros, es decir, el tiempo final de uno no debería ser mayor que el tiempo inicial del siguiente subtítulo. Sin embargo, esto puede ocurrir.
Asuma, además, que el archivo tiene muchos subtítulos, no sólo los del ejemplo de la derecha.

```
S1
00:00:20,000 --> 00:00:24,400
Te voy a pillar Bart

S2
00:00:24,100 --> 00:00:27,800
Maldito demonio

S3
00:00:29,100 --> 00:00:30,651
No Homero ... grrr
```

los_simpsons.srt

*a*) Escriba la función `tiempo_a_tupla(tpo)` que reciba un tiempo como texto y lo retorne como la tupla correspondiente.

```
>>> tiempo_a_tupla('00:00:20,000')
(0,0,20,0)
```

*b*) Escriba la función `solapados(nombre_archivo)` que reciba como parámetro el nombre del archivo de subtítulos y retorne una lista de tuplas con los subtítulos solapados. Cada tupla debe indicar los 2 subtítulos solapados.

```
>>> solapados('los_simpsons.srt')
[('S1', 'S2')]
```

(En el ejemplo hay sólo uno, pero podrían haber más)

*c*) Los subtítulos son una buena fuente para extraer los diálogos para una obra. Escriba la función `transformar_dialogo(nombre_archivo)` que reciba como parámetro el nombre del archivo de subtítulos y a partir de este cree un archivo `NOMBRE.dlg` (`NOMBRE` es el nombre del archivo antes del `.srt`) con los diálogos.

Al principio de cada línea se deben agregar 5 guiones bajos y dos puntos `'_____:'` (ahí se supone que posteriormente se escribirá el nombre del personaje que habla ese diálogo).
En la siguiente llamada se crea el archivo `los_simpsons.dlg` (ver a la derecha) a partir de los datos del archivo `los_simpsons.srt`.

```
_____: Te voy a pillar Bart
_____: Maldito demonio
_____: No Homero ... grrr
```

los_simpsons.dlg

```
>>> transformar_dialogo('los_simpsons.srt')
```