Programación—C.Recuperativo (CC) - Lunes 18 de Enero de 2016

Nombre: _____ Rol: _____ – _____

1. [25 %] Realice el ruteo de los siguientes programas e indique qué es lo que imprimen. Cada vez que el valor de una variable cambie, escríbalo en una nueva fila de la tabla. Recuerde que si una variable es de tipo string, debe colocar su valor entre comillas simples ' '. Si una variable almacena una función, coloque el nombre de ésta como valor (sin comillas).

***Importante**: La tabla tiene suficientes filas.*

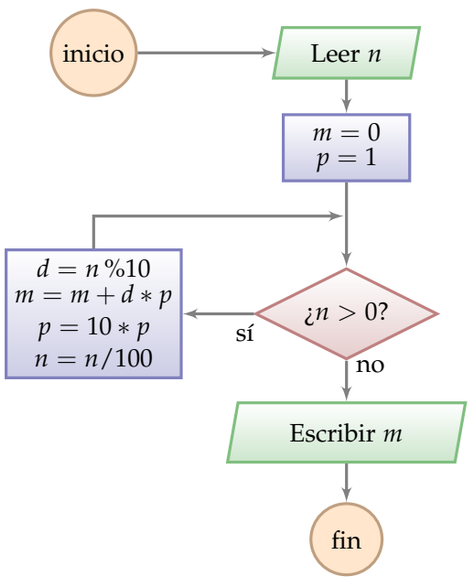
```
def geo(a, r):
    return a*r

def Sn(f, a, n = 5):
    s = 0
    for i in range(n):
        s += a
        a = f(a, 2)
    return s

r = 1
s = Sn(geo, r, 3)
print 'Total:', s
```

global		Sn					geo	
s	r	f	a	n	s	i	a	r

Análisis de Algoritmo

Analice el siguiente algoritmo y determine, **en pocas palabras**, lo que realiza. No debe usar más del espacio indicado.

Nota: Asuma la división como división entera.



Programación—C.Recuperativo (CC) - Lunes 18 de Enero de 2016

Nombre: Rol: –

2. **[35 %]** La gasolinera progra-Py-power ha solicitado un estudio sobre el uso de `N` surtidores de combustible a lo largo del tiempo. Cada día se registran las ventas (litros por auto) para cada surtidor, estos datos son almacenados en un diccionario como el siguiente:

```
# surtidor: [(litros venta, dia), ...],
gasolinera = { 1: [(43.3, 1), (28.2, 3)], 2: [(45.0, 2), (34.9, 3),
   (41.5, 4)], 6: [(16.0, 2), (21.5, 5), (45.0, 2)], 5: [], 4: [(5.2,
   10)], 9: [(15.2, 1), (2.5, 1)], # ...
}
```

Además existe un conjunto de tuplas con el surtidor y el tipo de bencina que contiene:

```
surtidores = {(4, 'Py-power'), (9, 'diesel'), (2, '95'), (1, '93'), (6,
    '97'), (5, 'Py-power'), # ...
}
```

Se le pide crear funciones para cumplir con los siguientes requerimientos:

a) Escribe la función `diccionario_bencinas(surtidores)` que genere un diccionario con llave tipo bencina y valor una lista con los surtidores que la proveen.

```
>>> diccionario_bencinas(surtidores)
{'93':[1], 'diesel':[9], '95':[2], 'Py-power':[4, 5], '97':[6]}
```

b) Escriba la función `mayor_beneficio(gasolinera, surtidores, precios)` que retorne el tipo de bencina que generó mayores ingresos por concepto de ventas. Considere el parámetro `precios` como un diccionario con el precio por litro de cada tipo de bencina. En caso de empate, regrese cualquiera de los tipos empatados.

```
>>> p= {'93': 38, 'diesel': 54, '95': 41, 'Py-power': 49, '97': 45}
>>> mayor_beneficio(gasolinera, surtidores, p)
'95'
```

c) Escriba la función `litros_vendidos(gasolinera, surtidores, dia, tipo)` que retorne los litros vendidos de un `tipo` de bencina en un `dia` dado.

```
>>> litros_vendidos(gasolinera, surtidores, 2, '97')
61.0
```

Programación—C.Recuperativo (CC) - Lunes 18 de Enero de 2016

Nombre: _____ Rol: _____ – _

3. **[40 %]** Una periodista del diario electrónico ´´El Encubridor" se ha conseguido un archivo con información de los viajes de los Senadores de Chile. Cada línea del archivo posee el rut de un senador y a continuación tríos con el mes (número), el país de destino y el *índice del costo*. El *índice del costo* es un número entero que identifica el número de la línea en otro archivo con el costo asociado al viaje. Los siguientes archivos son un ejemplo de lo anterior:

viajes.txt

```
9453454-6#10,Inglaterra,1;10,India,4
654676-3#10,Francia,5
6546334-4#3,Cuba,3
8764564-7#5,Francia,2
```

costos.txt

```
1 9000000
2 10000000
3 1020100
4 1500000
5 10000000
```

En el ejemplo, el senador-rut `654676-3` viajó a `Francia` el mes `10` a un costo de $10.000.000. Asuma que los ruts en el archivo de viajes no se repiten.

El editor en jefe le pide a un sansano en práctica que haga algunas funciones en Python para recopilar información más específica de estos personajes para así publicar un artículo de alto impacto en la edición del diario del día domingo.

a) Se necesita saber el total gastado por los senadores en viajes, por lo que se requiere la función `costo_total(arch)` donde `arch` es un archivo tipo `costos.txt`. La función entrega el costo total gastado por todos los senadores.

```
>> costo_total('costos.txt')
31520100
```

b) Se requiere la función `busqueda_por_mes(arch, mes)`, donde `arch` es un archivo tipo `viajes.txt` y `mes` el mes del viaje. Es necesario que la función retorne un diccionario con el rut del senador como llave y como valor una lista con los países de destino de aquellos políticos que viajaron en ese `mes`. Si no viajo nadie en ese periodo se retorna un diccionario vacío.

```
>> busqueda_por_mes('viajes.txt', 10)
{'9453454-6': ['Inglaterra', 'India'], '654676-3': ['Francia']}
```

c) Existe una ley que no permite a los senadores sobrepasar los montos totales iguales o superiores a $10.000.000, por lo que se quiere crear un archivo `sumario.txt`, en donde se guarde la información de los senadores que no cumplan dicha ley en el formato `Rut,Monto_total,destinos`, donde `destinos` es, a su vez, una serie de destinos separados por `';'` (ej: `destino1;destino2;...;destinoN`). Ahora se debe crear la función `realizar_sumario(arch1, arch2)`, donde `arch1` es un archivo del tipo `viajes.txt` y `arch2` un archivo tipo `costos.txt`. Toda la información debe ser guardada en un archivo llamado `sumario.txt`.

```
>>> realizar_sumario('viajes.txt', 'costos.txt')
>>>
```

sumario.txt

```
9453454-6,10500000,Inglaterra;India
654676-3,10000000,Francia
8764564-7,10000000,Francia
```