Universidad Simón Bolívar
Departamento de Computación y Tecnología de la información
CI-2691- Laboratorio de algoritmos I

Pre Laboratorio 3

El objetivo de este pre laboratorio es traducir algoritmos dados en GCL a Python y estudio de iteraciones.

Contenido: Iteraciones, ejemplo, ejercicios

Iteraciones

En GCL las iteraciones tienen la siguiente sintaxis:

do $B_0 \rightarrow S_0$
[] $B_1 \rightarrow S_1$
[] ...
[] $B_n \rightarrow S_n$
od

donde, **do** y **od** son palabras reservadas, B_i, $0 \leq i \leq n$, es una expresión booleana (o guardia) y su evaluación resulta en verdad o en falso, y S_i, $0 \leq i \leq n$, es una instrucción. Cada $B_i \rightarrow S_i$ es un comando con guardia, cuya guardia es B_i. La interpretación operacional de la instrucción iterativa es la siguiente:

Se evalúan todas las guardias. Si todas las guardias son "falso" (esto equivale a la condición de terminación), entonces se ejecuta la instrucción **skip**. En caso contrario, se escoge una guardia con valor "verdad" y se procede a ejecutar la instrucción correspondiente a esa guardia; una vez ejecutada la instrucción, se procede a ejecutar la instrucción iterativa nuevamente.

Ejemplo de Iteración en GCL y traducción a Python

A continuación se muestra el ciclo completo de la iteración en GCL:

```
i,suma:=0,0;
{ N > 0 ∧ 0 <= i <= N }
do i <= N ->
   if (i mod 2 = 0) ->
      suma:=suma + i
   [] (i mod 2 !=0) ->
      skip
   fi;
   i:=i+1
od
{ N > 0 ∧ i > N }
```

Donde $B_i : i \leq N$

La traducción a Python de este ciclo sería:

```
i,suma=0,0

# Verificación de precondición al inicio del ciclo
assert( N > 0 and 0<=i<= N)

while ( i <= N ):
   if (i % 2 == 0):
      suma=suma+i
   else:
      pass
   i=i+1

 # Verificación de post condición al salir del ciclo
 assert( N > 0 and i > N )
```

Como en Python no es obligatorio el uso del else, es decir, es opcional, el código puede simplificarse de la siguiente forma:

```
        i,suma=0,0

        # Verificación de precondición al inicio del ciclo
        assert( N > 0 ∧ 0<=i<=N)

        while ( i <= N ):
           if (i % 2 == 0):
              suma=suma+i
           i=i+1

        # Verificación de post condición al salir del ciclo
         assert( N > 0 ∧ i > N )
```

El ejemplo completo puede verlo y probarlo usando el archivo Lab3Ejemplo1.py adjunto a este prelaboratorio.

Ejercicios: Dadas los siguientes algoritmos en GCL, escriba un programa equivalente en Python. Para el ejercicio 1 use el pase de argumentos usando la función *input()* y en el ejercicio 2 use el método de pase de argumentos a través de la línea de comandos (ver instrucciones en PaseP-arametrosPython.pdf)

1) Prelab3Ejercicio1.py: algoritmo que calcula la suma de los factoriales desde 0 hasta N. Rcuerde que 0!=1.

```
[
        const N: int;
        var suma: int;
        var fact: int;
        var k: int;

        {N >= 0}
        suma,fact,k := 0,1,0;

        { N >= 0 ∧ 0<=k<=N }
        do ( k<=N ) ->
               if ( k > 0 ) ->
                      fact := fact * k
               []
                      skip;
               fi;
               suma := suma + fact;
               k := k + 1
        od
        { suma = (∑i: 0<=i<=N: (∏j: 1<=j<=i: j)) }
]
```

Donde el factorial se puede definir en función de la productoria $\prod$:

```
def prod( iterable ):
      p= 1
      for n in iterable:
            p *= n
      return p
```

Por ejemplo: `prod (j for j in range(1,N))` calcula N!

2) Prelab3Ejercicio2.py: algoritmo que calcula la suma de los dígitos de un número entero N de 10 dígitos máximo.

```
[
        const N: int;
        var suma: int;
        var cociente: int;

        {N > 0}

        suma,cociente := 0,N;

        do ( cociente > 0 ) ->
                suma := suma + cociente mod 10
                cociente := cociente div 10
        od

        {suma = ∑i: i = (N div 10^k) mod 10: (0 <= k <=10) ∧ (N div 10^k != 0)}
]
```

Condiciones de la entrega

Cree un archivo comprimido del tipo “tgz” llamado PreLab3-X.tgz y súbalo en el espacio del aula virtual de cada integrante del equipo antes de las 8:00 am del martes 4 de octubre. El archivo PreLab3-X.tgz debe contener los programas Prelab3Ejercicio1.py y Prelab3Ejercicio2.py y X es el número de carné de ambos integrantes.