Teoría de la Computación

Proyecto 2

Mario de León, 19019

FUNCIONES LAMBDA

main.py

```
"""
Funciones Lambda
Mario de Leon 19019
Teoria de la Computacion

a) cero(f,n)
b) uno(f,n)
c) dos(f,n)
d) tres(f,n)
e) sucesor(x,f,n)
f) suma(a,b,f,n)
g) multiplicacion(a,b,f,n)
h) potencia(a,b,f,n)
"""
# Funcion n = 0. No hay necesidad de f.
cero = lambda n: n # 0
# Funcion n = 1
# uno = lambda f, n: f(n)
uno = lambda f: lambda n: f(n) # 0+1
# Funcion n = 2
# dos = lambda f, n: f(f(n))
dos = lambda f: lambda n: f(f(n)) # 0+1+1
# Funcion n = 3
# tres = lambda f, n: f(f(f(n)))
tres = lambda f: lambda n: f(f(f(n))) # 0+1+1+1
# Funcion n = 4
# cuatro = lambda f, n: f(f(f(f(n))))
cuatro = lambda f: lambda n: f(f(f(f(n)))) # 0+1+1+1+1
# Funcion n = 5
# cinco = lambda f, n: f(f(f(f(f(n)))))
cinco = lambda f: lambda n: f(f(f(f(f(n))))) # 0+1+1+1+1+1

##############################################

# Funcion alpha
alpha = lambda n: n+1

# Funcion beta
beta = lambda n: n*2

##############################################

# Funcion sucesor(x,f,n): x (numero lambda), funcion, n (numero). Funcion n+1.
# suc = lambda x, f: lambda n: f(x(f), n)
```

```
---------- FUNCIONES LAMBDA ----------
Alpha: n+1
Alpha cero:  0
Alpha uno:  1
Alpha dos:  2
Alpha tres:  3
Alpha cuatro:  4
Alpha cinco:  5
Alpha sucesor de Uno:  2
Alpha suma de Uno + Dos:  3
Alpha multiplicacion de Dos * Tres:  6
Alpha potencia de Dos ^ Tres:  8

Beta: n*2
Beta cero, en realidad uno, ya que no es posible trabajar con cero:  1
Beta uno, en este caso 1*2:  2
Beta dos, 2*2:  4
Beta tres, 4*2:  8
Beta cuatro:  16
Beta cinco:  32
Beta sucesor, es decir Uno (2) * 2:  4
Beta suma de Uno + Dos, es decir, 2 * 4 debido a que beta es n*2:  8
Beta multiplicacion de Dos * Tres, es decir (4 * 8) * 2:  64
Beta potencia de Dos ^ Tres:  256
```

Funciones Alpha y Beta en los resultados

Se utilizó Alpha con argumento 0. Al momento de operar con 0, las operaciones Lambda retornan el resultado esperado. Por ejemplo, cero = 0, uno = 1, etc. Además, las operaciones matemáticas presentadas en el proyecto también retornan los resultados esperados. Ahora bien, con la función beta, los resultados son un tanto distintos de lo esperado. Al tratarse de una multiplicación x*2, los resultados de Beta van a ser alterados, y no los esperados. Por ejemplo, ya que beta es una multiplicación, los resultados van a crecer en múltiplos de 2. No se puede trabajar con el argumento 0. Como se observa en los resultados, uno va a utilizar la operación x*2, y si x = 1, entonces el número uno va a tener un valor de 2. Esto ya demuestra una inconsistencia si se desea trabajar con un número 1.

Para obtener los mejores resultados, es mejor usar la función Alpha con el argumento deseado “n” (reemplazando el default 0). De esta manera se pueden obtener resultados deseados y correctos para las operaciones matemáticas presentadas, convirtiendo al programa en una simple, pero útil, calculadora.