Error-tipico-con-listas

November 4, 2017

0.1 Discusión del error

Cuando cambiamos L en la línea L[0] += 1 cambian todas las apariciones de L en el código. Esto se puede evitar pasando a tuplas, que son inmutables, o copiando la lista antes y después de modificarla.

```
In [1]: def pruebita(n,N):
            L = [0]*n
            L1 = [L]
            for int in xsrange(N):
                L[0] += 1
                L1.append(L)
            return L1


In [2]: pruebita(2,10)

Out[2]: [[10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0]]

In [3]: def pruebita2(n,N):
            L = [0 for muda in srange(n)]
            L1 = [L]
            for int in xsrange(N):
                print L,L1
                L[0] += 1
                L1.append(L)
            return L1

In [4]: pruebita2(2,10)
```

```
[0, 0] [[0, 0]]
[1, 0] [[1, 0], [1, 0]]
[2, 0] [[2, 0], [2, 0], [2, 0]]
[3, 0] [[3, 0], [3, 0], [3, 0], [3, 0]]
[4, 0] [[4, 0], [4, 0], [4, 0], [4, 0], [4, 0]]
[5, 0] [[5, 0], [5, 0], [5, 0], [5, 0], [5, 0], [5, 0]]
[6, 0] [[6, 0], [6, 0], [6, 0], [6, 0], [6, 0], [6, 0], [6, 0]]
[7, 0] [[7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0]]
[8, 0] [[8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0]]
[9, 0] [[9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0]]
```

```
Out[4]: [[10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0],
         [10, 0]]
```

```
In [5]: def pruebita3(n,N):
            L = [0 for muda in srange(n)]
            L1 = [tuple(L)]
            for int in xsrange(N):
                print L,L1
                L[0] += 1
                L1.append(tuple(L))
            return L1
```

```
In [6]: pruebita3(2,10)
```

```
[0, 0] [(0, 0)]
[1, 0] [(0, 0), (1, 0)]
[2, 0] [(0, 0), (1, 0), (2, 0)]
[3, 0] [(0, 0), (1, 0), (2, 0), (3, 0)]
[4, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0)]
[5, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0)]
[6, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0)]
[7, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0)]
[8, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0), (8, 0)]
[9, 0] [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0), (8, 0), (9, 0)]
```

```
Out[6]: [(0, 0),
         (1, 0),
```

```
          (2, 0),
          (3, 0),
          (4, 0),
          (5, 0),
          (6, 0),
          (7, 0),
          (8, 0),
          (9, 0),
          (10, 0)]
```

```
In [7]: def pruebita4(n,N):
            L = [0 for muda in srange(n)]
            L1 = []
            for int in xsrange(N):
                print L,L1
                L2 = copy(L)
                L2[0] += 1
                L1.append(L2)
                L = copy(L2)
            return L1
```

```
In [8]: pruebita4(2,10)
```

```
[0, 0] []
[1, 0] [[1, 0]]
[2, 0] [[1, 0], [2, 0]]
[3, 0] [[1, 0], [2, 0], [3, 0]]
[4, 0] [[1, 0], [2, 0], [3, 0], [4, 0]]
[5, 0] [[1, 0], [2, 0], [3, 0], [4, 0], [5, 0]]
[6, 0] [[1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0]]
[7, 0] [[1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0], [7, 0]]
[8, 0] [[1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0], [7, 0], [8, 0]]
[9, 0] [[1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0], [7, 0], [8, 0], [9, 0]]
```

```
Out[8]: [[1, 0],
         [2, 0],
         [3, 0],
         [4, 0],
         [5, 0],
         [6, 0],
         [7, 0],
         [8, 0],
         [9, 0],
         [10, 0]]
```

```
In [9]: def pruebita5(n,N):
            L1 = [[0]*n]
            for int in xsrange(N):
```

```
            L2 = copy(L1[-1])
            L2[0] += 1
            L1.append(L2)
        return L1
```

```
In [10]: pruebita5(2,10)
```

```
Out[10]: [[0, 0],
          [1, 0],
          [2, 0],
          [3, 0],
          [4, 0],
          [5, 0],
          [6, 0],
          [7, 0],
          [8, 0],
          [9, 0],
          [10, 0]]
```

Si se asigna sin el copy, en la forma L2 = L1[-1], se está cambiando también en la lista de listas $L1$. El motivo es que $L2$ es una variable que se refiere a las mismas posiciones de memoria que $L1[-1]$, y cualquier cambio hecho sobre $L2$ afecta a $L1$.

```
In [11]: def pruebita6(n,N):
             L1 = [[0]*n]
             for int in xsrange(N):
                 L2 = L1[-1]
                 print L2
                 L2[0] += 1
                 print L2
                 L1.append(L2)
                 print L1
                 print ("====================================")
             return L1
```

```
In [12]: pruebita6(2,10)
```

```
[0, 0]
[1, 0]
[[1, 0], [1, 0]]
====================================
[1, 0]
[2, 0]
[[2, 0], [2, 0], [2, 0]]
====================================
[2, 0]
[3, 0]
[[3, 0], [3, 0], [3, 0], [3, 0]]
====================================
```

```
[3, 0]
[4, 0]
[[4, 0], [4, 0], [4, 0], [4, 0], [4, 0]]
==================================
[4, 0]
[5, 0]
[[5, 0], [5, 0], [5, 0], [5, 0], [5, 0], [5, 0]]
==================================
[5, 0]
[6, 0]
[[6, 0], [6, 0], [6, 0], [6, 0], [6, 0], [6, 0], [6, 0]]
==================================
[6, 0]
[7, 0]
[[7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0], [7, 0]]
==================================
[7, 0]
[8, 0]
[[8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0], [8, 0]]
==================================
[8, 0]
[9, 0]
[[9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0], [9, 0]]
==================================
[9, 0]
[10, 0]
[[10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10, 0], [10
==================================
```

Out[12]:
```
[[10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0],
 [10, 0]]
```

In [ ]: