54-COMPL-eficiencia-ram

December 3, 2017

```
In [1]: get_memory_usage()

Out[1]: 907.87890625

In [2]: def prueba_ram():
            import gc # Para poder usar la linea 8
            L = [get_memory_usage()]
            for n in srange(3,10):
                A = "B"*10^n #Una cadena de caracteres de longitud 10^n
                L.append(get_memory_usage())
                del A                           #Borramos A
                gc.collect()                    # Recogemos la basura generada por A
                L.append(get_memory_usage())
            return L

In [3]: prueba_ram()

Out[3]: [907.87890625, 907.87890625, 907.87890625, 907.87890625, 907.87890625, 907.87890625,
         907.87890625, 908.8359375, 907.87890625, 917.41796875, 907.87890625, 1003.25,
         907.87890625, 1861.5546875, 907.87890625]
```

£Por qué inicialmente no se incrementa la memoria RAM en uso? Debe ser que el sistema reserva una cierta cantidad de RAM, aparentemente en esta máquina 907 MB, y mientras no necesita superar esa cantidad no vemos aumentos. Una vez que la estructura de datos que estamos construyendo necesita una cantidad de RAM que supera la reservada empezamos a ver los incrementsos en el resultado.

Al construir la cadena con longitud 10^9, la última, el incremento sobre la memoria reservada es de 953.67578125 MB, que son $(953.6757812) * 1024^2 = 1.00000153600000 * 10^9$ bytes, que es lo correcto porque cada caracter ocupa un byte, es decir 8 bits.

```
In [4]: def prueba_ram2():
            import gc
            L = [get_memory_usage()]
            A1 = ["ABCDEFGHIJ"]
            for n in srange(3,9):
                A = (10^n)*A1       #Ahora A1 es una lista y A tiene 10^n copias de A1
                L.append(get_memory_usage())
```

```
            del A
            gc.collect()
            L.append(get_memory_usage())
        L.append(get_memory_usage())
        return L
```

```
In [5]: prueba_ram2()
```

```
Out[5]: [907.8828125, 907.8828125, 907.8828125, 907.8828125, 907.8828125, 908.6484375,
         907.8828125, 915.515625, 907.8828125, 984.1796875, 907.8828125, 1670.82421875,
         907.8828125, 907.8828125]
```

Es curioso que una lista con 10^8 veces la cadena "ABCDEFGHIJ" parece ocupar menos que la cadena de longitud 10^9 del primer ejemplo.

Ejercicio

£Cómo estimarías la cantidad de RAM que utiliza la lista $srange(N)$? Comprueba tu estimación definiendo, como en los ejemplos anteriores, una función adecuada.