Práctico #3

Test Automation With Python Noviembre 2016.

Forma de entrega: Via Email a jbc.develop@gmail.com

Se evalúa:

- Cantidad de características implementadas
- PEP 8 (con flake8 3.0)
- PEP 20: Zen de Python
- Uso de características de Python

Entregar Jueves 15 de diciembre del 2016

Ejercicio 1

Implementar un decorador `@thread` que al decorar una funcion hace que la ejecución de la misma sea hecha en un thread aparte.

El cambio fundamental es que la función decorada no retorna lo mismo que la función base sino que retorna un objeto `Promise`. Promise es-un *thread* donde se ejecuto la función original, y además posee un método `get_result()` que retorna el valor final de la función que falla en caso del que thread no haya terminado.

Ejemplo:

```
import time

@concurrent
def function(a, b):
    # esta funcion duerme aprox 10 segundos ante de terminar
    time.sleep(10)
    return a + b

promise = function(1, 2)

# falla si se llama antes de ~10 segundos o devueltve 3
promise.get_result()
```

Ejercicio 2

Implementar un decorador `@jsonparams` que permite que una funcion cualquiera acepte sus parametros en forma de un string JSON.

```
@jsonparams
def function(a, b):
    return a + b


function('{"a": 1, "b": 2}') # retorna 3
function('[1, 2]') # retorna 3
```

Ejercicio 3

Implementar un script que reciba argumentos por linea de comando con *argparse*.

Dicho script recibe por parámetro dos fechas-y-hora en formato ISO (https://docs.python.org/2/library/datetime.html#datetime.datetime.isoformat) y retorna su diferencia en segundos.

Los parametros **OBLIGATORIOS** de dicho script son:

- `--from` fecha de inicio.
- `--to` fecha de fin.

Ejemplo 1: ambos parámetros son obligatorios

```
$ python ejemplo.py
usage: ejemplo.py [-h] --from DATE --to DATE
ejemplo.py: error: argument --from is required
```

Ejemplo 2:

```
$ python ejemplo.py --from 2016-12-06T21:17:07.886069 --to 2016-12-06T21:17:19.053350
Diferencia: -11.167281 secs.
```