Paradigmas de la programación

Practica 2.

Por Javier Cámara Jabonero

Función es_palindromo

```
def es_palindromo(list):
    true_palindromo=map(lambda x:True if x[::-1]==x else False, list)
    return(true_palindromo)
```

Esta función recibe una lista, ya puede ser de una o varias palabras y comprueba que sean palíndromos. En la salida devuelve otra lista la cual es del mismo tamaño que la inicial con True o False, dependiendo de si son o no palíndromos. La función funciona en base a un map, el cual recibe una función, en este caso una lambda y una lista donde estarán nuestras palabras. La función lambda lo que hará será si la variable x es igual a su reversa (x[::1] es la reversa) nos introducirá en la lista el bool true, en caso contrario false.

Función impares_de

```
def impares_de(list):
    vector_impares = filter(lambda x: x%2==0,list)
    return(vector_impares)
```

La función impar recibirá un vector de números y los filtrara dependiendo de si son impares o no. En este caso como solo queremos saber que números son impares la función filter eliminara los elementos de la lista que sean pares. Al igual que el map está recibe un una función y una lista, la función que he usado aquí es una lambda, que solo recoge los números que al dividirlos entre 2 dan un resto 0, ósea los pares. La salida de la función es el vector de números impares.

Función cuadrados_sumados

```
def cuadrados_sumados(n):
    suma_cuadrados=reduce(lambda acc,suma:acc+pow(suma,2),range(1,n+1))
    return(suma_cuadrados)
```

La función cuadrados sumados recibe en este caso un entero n, el cual es el número de iteraciones que queremos que haga. En este caso utilizare una función reduce la cual recibe una función y una lista, como no tengo en este caso una lista he creado un rango entre 1 y n+1 (el rango en Python va de 0 a n-1) y la función lambda en este caso tiene dos variables, la primera será la que devolverá el reduce, el total de todas las sumas, y la segunda es el iterador del rango. Finalmente, la función devuelve un int el cual es el total de sumatorio.

Función factorial

```
def factorial(n):
    fact_num = reduce(lambda acc,suma:acc*suma,range(1,n+1))
    return(fact_num)        Find related code in Paradigmas-2
```

La función factorial recibe un entero, esta he usado la misma forma que en la de cuadrados sumados, pero en vez de hacer un sumatorio en esta he de hacer un productorio, el cual va igual que en cuadrados sumados, a través del rango de números. También devuelve un int con el total de la factorial.

Función es_primo

```python
def es_primo(n):
    primo= filter(lambda x: n % x ==0 ,range(1,int(n/2)+1))
    return(False if len(list(primo)) > 1 else True)
```

Para la función es primo recibe un entero el cual queremos saber si es primo o no, esta nos devuelve true o false. La función que he usado aquí es un filter el cual recibe una lambda la cual va de 1 a n/2 +1 lo que sería n medios. La función devolverá una lista la cual contiene todos los divisores de n, si la lista contiene un valor superior a 1 significa que tiene más de un divisor por lo tanto no es primo, e viceversa. La función devuelve un bool.

Finalmente, esta son las llamadas a mi función y los prints que he hecho.

```python
import function as f

def main():
    palindromos = ["ana","pepe","otto","carmen","reconocer","granaino","orejero","hugo","javi","leila"]
    print(palindromos)
    palin=f.es_palindromo(palindromos)
    print(list(palin))
    impares=[1,22,34,43,55,6,123,8,90,120]
    print(impares)
    impares=f.impares_de(impares)
    print(list(impares))
    n=12        Find related code in Paradigmas-2      You, 2 hours ago • toda la practica
    cuadrados_sumados=f.cuadrados_sumados(n)
    print("el cuadrado sumado de {num} es {cuadrado}".format(num=n,cuadrado=cuadrados_sumados))
    n=10
    fact=f.factorial(n)
    print("el factorial sumado de {num} es {fact}".format(num=n,fact=fact))
    n=3110
    primo=f.es_primo(n)
    if primo == False:
        print("El numero {num} no es primo".format(num=n))
    else:
        print("El numero {num} es primo".format(num=n))
```