

PARCIAL CORTE 2
ELECTIVA DE PROGRAMACION

DOCUMENTACION DE LOS PROGRAMAS

PRESENTA JUAN DAVID JIMENEZ

CODIGO ESTUDIANTE: 71101

- Punto No. 1 Valor del punto 1 = 1.0
 - Punto No. 2 Valor del punto 2 = 1.5
 - Punto No. 3 Valor del punto 3 = 2.5
-
- 1.Escriba un programa que tenga como entrada los números de la serie de Fibonacci. El programa debe calcular el acumulado de los números para un rango dado por el usuario así: Si el número a computar es primo lo debe sumar al valor acumulado y si no es primo lo debe elevar al cuadrado y sumarlo al acumulado. Utilice funciones como MAP, filter y reduce. Punto No. 1 Valor del punto 1 = 1.0
 - Como se abordó el problema:
 -
 - Para empezar, utilice como base las listas, porque se me ocurrió que podría almacenar los datos que fuese recopilando cada función o cada filtro que realizara una función lambda que se realizara para poder mover los elementos más fácilmente.
 - En este problema analice que por medio de la consola tendría que recibir los valores del rango desde donde se quiere que se empiece la serie de Fibonacci, siguiendo en cómo crear una serie de Fibonacci que estuviera en una lambda, en la cual pudiera trabajarse desde el máximo rango dado por el usuario.

- Función en la cual me recopila los datos ingresados por el usuario.

```
def number():
    while True:
        user_input = input('Ingrese un numero mientras que no sea negativo rango inicial: ')
        user_input_2 = input('Ingrese el rango final:')
        if user_input.isnumeric() and user_input_2.isnumeric(): #Si los 2 se cumplen , retorna los elementos que tienen .i
            return int (user_input), int (user_input_2)
        else:
            print("Error , continuar")
            continue
```

- Función lambda para la serie de Fibonacci, me di cuenta de que es mucho más práctico generar una función lambda para la serie, que trabajar por una función convencional que tuviese sentencias if o else ya que recibiría los n elementos para la serie, ello hace que la sintaxis sea mucho más práctica.
- Función lambda para serie Fibonacci.
- Para crear la serie hay que tener en cuenta que los n elementos ingresados, es el conjunto de valores a operar, los elementos raíz de mi serie son el 0 y 1.
- El rango me lo digita el usuario, como generar_serie_fibo() me retornara un cálculo de la serie, por ello uso la función de reduce() Esta función se utiliza para llevar a cabo un cálculo acumulativo sobre una lista de valores y que me devolverá el resultado.

```
#Genera serie de fibonacci
generar_serie_fibo = lambda x: reduce(lambda n, _: n + [n[-1] + n[-2]], range(x - 2) , [0, 1])
```

- Ya después de generar la serie a partir del rango dado, me puse a analizar de cómo podría filtrar los elementos de mi serie, como dice en el problema que debemos de decir que elementos son primos y que otros no son primos, Implemente un filtro para cada caso.

- Para el caso de los primos tengo la siguiente función yes_primo().
- En la cual va a recibir los n elementos de una lista y esos elementos me determina si es primo o no.

```
def yes_primo(x):
    if x <= 1: #evaluamos e
        return False
    for i in range(2, x- 1):
        if x % i == 0:
            return False
    return True
```

-

- Cuando obtuve la función de primo aplique que podría usarse la función de filter() que ella dice que recibe una función y una lista, entonces lo siguiente que hice, fue crear una nueva variable para tener los primos en una lista aparte.

```
#Creamos variables con la finalidad de filtrar nuestras listas de numeros
primo = list(filter(yes_primo, fibo_list)) #filtramos la lista de la serie de fibonacci
```

- Después de tener la serie de Fibonacci filtrada, los números primos los tengo aparte en una lista, entonces analice un poco mas las cosas y obtuve lo siguiente si tengo los primos, puedo decirle a mi lista de Fibonacci que me retire los elementos que son iguales a la lista de los primos utilice funciones de has_Set además utilizando el operador de diferencia pude sacar los elementos que no son primos.

```
#Filtramos los elemntos de una lista de la otra que no son primos
lis1 = set(fibo_list)
lis2 = set(primo)
diferentes = lis1.difference(lis2)
```

- Siguiendo a ello, ya que puedo determinar mis resultados, me piden que los elementos que son primos se sumen y los elementos que no son primos se eleven al cuadrado. Como ya tengo 2 listas aparte que contienen los elementos filtrados puedo realizar lo siguiente.

- Función lambda no_primo.
- Como tengo ya la lista de los que no son primos, puedo calcular los números que no son primos al cuadrado por medio de la función map() ya que esta hace la ejecución de una función lambda que eleva al cuadrado en este caso y recibe una lista y me devuelve los valores calculados.

```
#Numero que no son primos
no_primo = list(map(lambda n: n**2 ,diferentes)) #aplicamos a la lista no primo la operacion de elevarlos al cuadrado
```

- Hasta ahora ya hemos resuelto nuestro problema, ahora solo nos falta es el calculo de los acumulados de nuestras listas primo y no_primo, por medio de la función sum() se realiza la suma de los elementos que contiene mi lista.

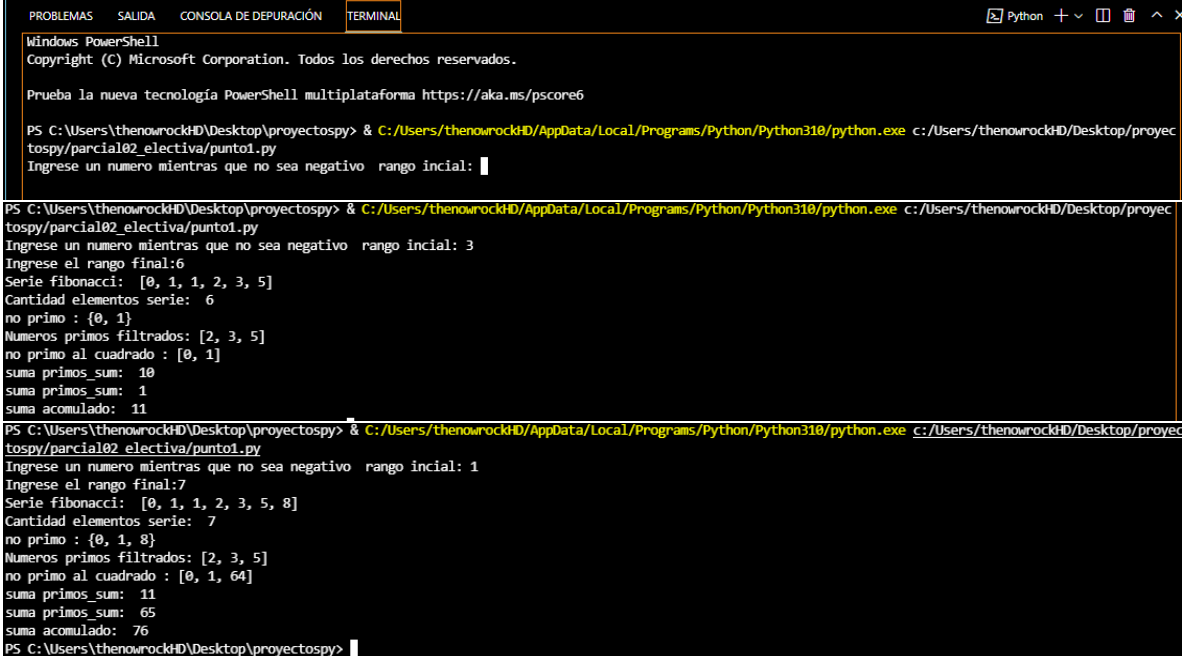
```
#Calcular la suma de nuestros elementos de la listas
primos_sum = sum(primo) #+ 1 #le sumo + 1 , ya que e
no_primo_sum = sum(no_primo) #sumamos el acumulado de
```

- Como parte final hace la suma total del acumulado entre primo y no_primo.

○

```
#Calculamos el acumulado de toda la serie de fibonacci
suma_acumulado = primos_sum + no_primo_sum #sumamos las listas de los primos y no _ primos que serian el acumulado total
```

- Pruebas de salida:

○ The screenshot shows a Windows PowerShell terminal window with the following content:
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.
Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6
PS C:\Users\thenowrockHD\Desktop\proyectospy> & C:/Users/thenowrockHD/AppData/Local/Programs/Python/Python310/python.exe c:/Users/thenowrockHD/Desktop/proyectospy/parcial02_electiva/punto1.py
Ingrese un numero mientras que no sea negativo rango inicial: 1
Ingrese el rango final: 6
Serie fibonacci: [0, 1, 1, 2, 3, 5]
Cantidad elementos serie: 6
no primo : {0, 1}
Numeros primos filtrados: [2, 3, 5]
no primo al cuadrado : [0, 1]
suma primos sum: 10
suma primos sum: 1
suma acumulado: 11
PS C:\Users\thenowrockHD\Desktop\proyectospy> & C:/Users/thenowrockHD/AppData/Local/Programs/Python/Python310/python.exe c:/Users/thenowrockHD/Desktop/proyectospy/parcial02_electiva/punto1.py
Ingrese un numero mientras que no sea negativo rango inicial: 1
Ingrese el rango final: 7
Serie fibonacci: [0, 1, 1, 2, 3, 5, 8]
Cantidad elementos serie: 7
no primo : {0, 1, 8}
Numeros primos filtrados: [2, 3, 5]
no primo al cuadrado : [0, 1, 64]
suma primos sum: 11
suma primos sum: 65
suma acumulado: 76
PS C:\Users\thenowrockHD\Desktop\proyectospy>

- No sé porque en este caso me da 75, debería de dar 72, no sé si debe de ser a la hora de filtrar los números primos de la serie, para que me queden los números que no son primos en una nueva lista. No sé porque me repite el primer elemento de la lista nueva creada, he buscado diferentes soluciones para esto y no encuentro por qué funciona así los sets (). encontré en un artículo de stack overflow que por lo regular este tipo de funciones generan un error en las listas cuando pasan los elementos. Eh que falla ya que tengo problemas con esto, no es para todos los casos.

- ```
PS C:\Users\thenowrockHD\Desktop\proyectospy> & C:/Users/thenowrockHD/AppData/Local/Programs/Python/Python310/python.exe c:\tospy\parcial02_electiva\punto1.py
Ingrese un numero mientras que no sea negativo rango inicial: 5
Ingrese el rango final:7
Serie fibonacci: [0, 1, 1, 2, 3, 5, 8]
Cantidad elementos serie: 7
no primo : {0, 1, 8}
Numeros primos filtrados: [2, 3, 5]
no primo al cuadrado : [0, 1, 64]
suma primos_sum: 10
suma no_primo_sum: 65
suma acumulado: 75
```

- 2. Escriba un programa en Python que utilizando expresiones lambda devuelva los números de la siguiente serie: Para generar los n números a calcular en la serie, utilice una función generadora de números enteros mayores a 0.
- Como se abordó el problema:
- En este problema lo primero que pensé es que podría hacer una función generadora en la cual me recibiera el rango de los números que el usuario quisiera hacer la serie dando un rango(inicio, final) en el que pudiera trabajarse, aquí implemente que en el rango dado podría iterarse y genera números aleatorios de ese rango para la serie.
- Primera parte diseñe las entradas de usuario que me indica un numero inicial que debe ser > 0 y de carácter entero.

- ```
#Input usuario , Rango inicial y rango final .
r_i = int(input('Ingrese el rango inicial para generar aletorios que sean > 0 :'))
r_f = int(input('Ingrese el rango final para generar aletorios que sean > 0 :'))
```

- Para generar los números que necesita mi serie, diseñé un generador de números aleatorios que me recibe el rango dado por el usuario y me genera la n cantidad de números que están entre el rango como resultado obtuve la siguiente función que la convierto en una lista, ya que se me hace más fácil manejar los datos con ella no es necesario convertir la función generadora en una lista, pero esto hace me hizo más fácil trabajar con ellos.
- Para poder generar los números aleatorios utilizamos la librería random, en la cual encontramos la funciones de random.randint(inicio, final) recibe 2 parámetros, que serian el rango de los números a generar.

- ```
#Funcion generadora de numeros aletorios, en el rango de n
#Yo aplique que los numeros generados podrian ser aletorios y en un rango dado.
randoms = list(random.randint(r_i,r_f) for r_i in range(r_i,r_f)) #Genera numeros apartir del r_i >= r_f y apartir de la
```

- Para calcular la serie dada, procuré mirar la manera mas eficiente de encontrar la factorial del n, buscando un poco encontré que podía realizar esta operación por medio de la librería de math, para hacer uso de la función de factorial de la librería math, se hace el llamado de math.factorial(n) y recibe como parámetro el numero que se le quiere aplicar la factorial.
- Se solicita a calcular la serie por medio de una función lambda en la cual me recibiera los números generados y que me calcule la serie de el numero ingresado el resultado fue.

```
serie = (lambda n: math.factorial(2*n) / (math.factorial(n + 1) * math.factorial(n)))
```

- Listo como ya tenemos los componentes para calcular la serie, procedemos a la aplicación de estas funciones diseñadas, por medio de calcular\_serie aplicamos la función de map(), map nos ayuda con la aplicación de una operación a una lista y nos devuelve su resultado, entonces nos queda de la siguiente manera, en que recibe la serie y los números generados.

```
calcular_serie = list((map(serie, randoms)))
```

```
PS C:\Users\thenowrockHD\Desktop\proyectospy> & C:/Users/thenowrockHD/AppData/Local/Programs/Python/Python310/python.exe c:/Users/thenowrockHD/Desktop/proyectospy/parcial02_electiva/punto2.py
Ingrese el rango inicial para generar aleatorios que sean > 0 :
Ingrese el rango inicial para generar aleatorios que sean > 0 :2
Ingrese el rango final para generar aleatorios que sean > 0 :10
Lista generada aleatoriamente : [5, 4, 9, 5, 7, 10, 8, 9] Serie calculada : [42.0, 14.0, 4862.0, 42.0, 429.0, 16796.0, 1430.0, 4862.0]
```

- Escriba un programa que basado en 3 palabras dadas por el usuario construya un crucigrama. El objetivo es maximizar en el crucigrama el número de letras en común entre las palabras. Condiciones: • El tamaño del crucigrama no está predeterminado y puede ser el resultado de la optimización solicitada. • El tamaño de las palabras no está predeterminado • La orientación de las palabras puede ser cualquier dirección horizontal o vertical.

- Como se abordó el problema:

Lo redacte en el documento punto3.py

PS C:\Users\thenowrockID\Desktop\proyectospy> & C:/Users/thenowrockID/AppData/Local/Programs/Python/Python310/python.exe c:/Users/thenowrockID/Desktop/proyectospy/parcial02\_electiva/punto3.py  
Digite una palabra:

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN **TERMINAL**

['CASA', 'MANO', 'PAZ']

C

M A N O

S

P A Z

Numero de letras comunes = 2

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN **TERMINAL**

['VENTA ', 'MATAR', 'ATAR']

V

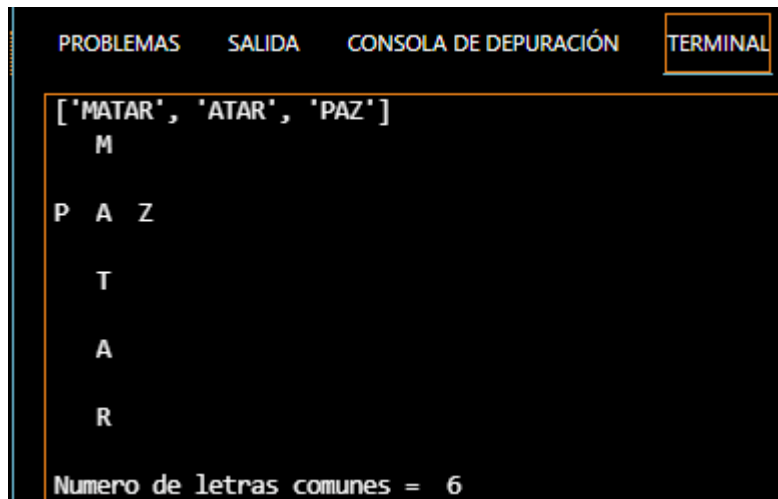
E

N

M A T A R

A

Numero de letras comunes = 6



Anexo esta excusa medica ya que tuve un accidente que me dejo mal sinceramente de mi cabeza y me dificulto terminar bien el punto 3 del parcial, ya que no lo tenia completo y termine con ayuda de mi compañero.

