Laboratorio1

September 10, 2020

MAT281 - 2° Semestre 2020

Profesor: Francisco Alfaro Medina

0.1 Problema 01

0.1.1 a) Calcular el número π

Al escribir la serie, la función queda de la siguiente manera:

[1]:
```
def Calculo_pi(n):
    suma = 0                 #En esta variable tendremos la serie
    for k in range(1,n + 1):
        suma = ((-1)**(k + 1))/(2*k - 1) + suma  #Escribimos la serie dada para␣
↪multiplicar por 4 al final
    return 4 * suma
```

[2]:
```
Calculo_pi(3)
```

[2]:
```
3.466666666666667
```

[3]:
```
Calculo_pi(100000)
```

[3]:
```
3.1415826535897198
```

0.1.2 b) Calcular el número e

Necesitamos una función para los factoriales `factorial()` presentes como denominadores en la serie, por lo tanto las funciones quedan de la siguiente manera:

[4]:
```
def factorial(numero):              #Esta funcion es para el factorial dado en␣
↪el denominador de la serie
    if numero == 1 or numero == 0:
        return 1
    else:
        return (numero * factorial(numero - 1))
```

```
def Calculo_e(largo):
    suma = 0                    #Esta sera la serie
    for k in range(0,largo):
        denominador = factorial(k)
        suma = 1/denominador + suma     #Vamos escribiendo la serie termino a␣
 ↪termino
    return suma
```

[5]:
```
Calculo_e(3)
```

[5]: 2.5

[6]:
```
Calculo_e(1000)
```

[6]: 2.7182818284590455

0.2 Problema 02

Como el problema se centra mayormente en la suma de sus divisores propios, se crea la función para obtener estos `suma_propios()`, y luego la función buscada la que nos dirá si son o no amigos `Amigos()`.

[7]:
```
def suma_propios(numero):    #Esta funcion es para devolver la suma de los␣
 ↪divisores de un numero
    if numero == 1:
        return 0
    divisores = []
    numero2 = int(numero/2 + 1)   #hasta la mitad del numero, ya que mas grande␣
 ↪no es divisible por este
    for k in range(1,numero2):
        if numero % k == 0:
            divisores.append(k)    #Agregamos a una lista y sumamos
    return sum(divisores)

def Amigos(numero1,numero2):
    if suma_propios(numero1) == numero2 and numero1 == suma_propios(numero2): ␣
 ↪#Finalmente vemos si se
        return True                                                          ␣
 ↪#cumple condicion de
    return False                                                              ␣
 ↪#numeros amigos
```

[8]:
```
Amigos(220,284)
```

[8]: True

[9]:
```
Amigos(6,5)
```

[9]:
```
False
```

0.3 Problema 03

La función de `Collatz()` queda de la siguiente manera:

[10]:
```
def Collatz(Numero):
    if Numero <= 1:
        return 1
    collatz = [Numero]          #Agregamos el primer numero a la lista
    while Numero != 1:
        if Numero % 2 == 1:              #Luego agregamos las operaciones dadas␣
↪para cada numero
            Numero = int(Numero * 3 + 1)  #que nos de hasta llegar al 1
            collatz.append(Numero)
        else:
            Numero = int(Numero / 2)
            collatz.append(Numero)
    return collatz
```

[11]:
```
Collatz(9)
```

[11]:
```
[9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]
```

0.4 Problema 04

Como la función `Goldbach()` es la suma de dos numeros primos, necesitamos verificar primero que los numeros que sumaremos son numeros primos o no, para eso está la función `esonoprimo()`, y al trabajar al final con solo numeros primos se busca la combinación que cumpla con el objetivo:

[12]:
```
def esonoprimo(numero):    #Esta funcion revisa si es o no primo el numero dado
    if numero < 2:          # 0 si no es primo
        return 0            # 1 si lo es
    elif numero == 2:
        return 1
    numero2 = int(numero / 2 + 1)
    for k in range(2,numero2):
        if numero % k == 0:
            return 0
    return 1

def Goldbach(Numero):
```

```
    if Numero <= 3:           #si el numero es menor o igual a 3 este no puede␣
↪ser suma de 2 primos
        return False
    primos = [k for k in range(2,Numero) if esonoprimo(k)==1]  #Agregamos los␣
↪numeros primos menores
    for elemento in primos:                                    #a nuestro␣
↪numero en cuestion
        for number in primos:
            if elemento + number == Numero:              #Vemos alguna␣
↪combinacion de numeros primos
                primos.append(elemento)                  #que de el numero en␣
↪cuestion
                primos.append(number)                   #Por lo dicho por␣
↪Goldbach
                return (primos[-2],primos[-1])          #esta condicion siempre␣
↪se cumple
```

[13]:
```
Goldbach(4)
```

[13]: (2, 2)

[14]:
```
Goldbach(6)
```

[14]: (3, 3)

[15]:
```
Goldbach(8)
```

[15]: (3, 5)

[16]:
```
Goldbach(1000)
```

[16]: (3, 997)