INFORME DE SUCESIONES ALÍCUOTAS

Integrantes: Tomas Aguirre, Menchaca Miguel Ángel, Paulo Sosa

Encargado de la entrega: Menchaca Miguel Ángel

Curso: 06

Profesor: A. Servetto

Fecha de entrega: 25/10/2021

Contenido

Consignas del Trabajo Practico

Desarrollar un programa que solicite al usuario números naturales mayores a 1 hasta que ingrese 0, que para cada número (excepto para el 0) imprima su sucesión alícuota o parte de ella en una misma línea (sin mostrar términos repetidos), y que, según el último término calculado de la sucesión, imprima al final de la misma línea si el origen de la sucesión es un número perfecto, ambicioso o primo, si la sucesión es de amigos o socios, si el penúltimo término de la sucesión es primo o si todos sus términos son presumiblemente abundantes. Tener en cuenta que el ciclo más grande conocido de números socios es de 28 números, por lo que si no se llega a un término igual al origen o al anterior o a la unidad, en el peor de los casos puede calcular a lo sumo 27 términos (sin contar el origen).

Para casos de análisis y prueba se puede desarrollar sucesiones alícuotas de:

1. 496 es perfecto
2. 220 284 son amigos
3. 12496 14288 15472 14536 14264 son socios
4. 14316 19116 31704 47616 83328 177792 295488 629072 589786 294896 358336 418904 366556 274924 275444 243760 376736 381028 285778 152990 122410 97946 48976 45946 22976 22744 19916 17716 son socios

5. 95 25 6 - 95 es ambicioso
6. 4274 2140 2396 1804 1724 1300 1738 1142 574 434 334 170 154 134 70 74 40 50 43 1 (sucesión finita con un primo como penúltimo término)
7. 552 888 1392 2328 3552 6024 9096 13704 20616 30984 46536 86904 165816 367704 628356 837836 628384 630356 491884 368920 499400 772840 978650 975652 744248 696712 628628 857836 857892 (más de 28 términos abundantes)

Requerimientos de la entrega

1. El problema debe resolverse en forma grupal, y debe realizar la entrega sólo un miembro en representación de su grupo (los integrantes del grupo deben figurar en la documentación del programa).

2. El problema debe solucionarse utilizando únicamente los recursos de Python conocidos hasta el momento de su planteo.

3. El programa debe ser eficaz (informar resultado esperado y cumplir con especificación del enunciado), inteligible (utilizar el modelo o plantilla y completar todas las secciones que sean pertinentes al problema) y eficiente (lograr calidad de diseño).

4. Junto con el programa se debe entregar un documento con un informe del desarrollo del trabajo, consignando integrantes del grupo, y, para cada reunión virtual o presencial de trabajo, descripción de objetivo, resultados, fecha, hora, modalidad (por ejemplo, Meet compartiendo pantalla) y duración. El formato del informe debe ser preferentemente PDF (se admite también RTF, DOCX, ODT o TXT).

Nomenclatura obligatoria de archivos para la entrega:

Alícuotas_Apellido1_Apellido2_Apellido3.py e ídem para el informe, con la extensión que corresponda

Reuniones:

- **1º REUNIÓN**

  Fecha:19/10/2021
  Objetivo: Hacer esquema y estructura del tp
  Resultados: Se nos presentaron dificultades expresando la sucesión, se intentaron dos modelos de los cuales elegimos uno para continuar, posterior a esto establecimos los condicionales bases y falta completar la sucesión
  Modalidad: Reunión virtual (Meet)
  Duración: 01 hs 45 ms

- **2º REUNIÓN**

  Fecha: 21/10/2021
  Objetivo: completar la sucesión y cumplir las pautas requeridas para la entrega del trabajo práctico
  Resultados: completamos el código de la sucesión pero se presentaron problemas para la expresión de términos por ende tuvimos que reestructurar el código y sus condicionales para dar las expresiones requeridas
  Modalidad: Reunión virtual (Meet)
  Duración: 01 hs 30 ms

- **3º REUNIÓN**

  Fecha: 23/10/2021
  Objetivo: Finalizar, repasar el cumplimiento y funcionamiento del trabajo práctico
  Resultados: Se pudo finalizar el trabajo y utilizamos todos los números de prueba para verificar el cumplimiento y eficacia del programa.
  Modalidad: Reunión virtual (Meet)
  Duración: 45 ms

Sección Declarativa:

'''

Objetivo del programa (descripción del problema que resuelve):

Desarrollar un programa que solicite al usuario números naturales mayores a 1 hasta que ingrese 0, que para cada número
(excepto para el 0) imprima su sucesión alícuota o parte de ella en una misma línea (sin mostrar términos repetidos), y que,
según el último término calculado de la sucesión, imprima al final de la misma línea si el origen de la sucesión es un número
perfecto, ambicioso o primo, si la sucesión es de amigos o socios, si el penúltimo término de la sucesión es primo o si todos sus
términos son presumiblemente abundantes.
Tener en cuenta que el ciclo más grande conocido de números socios es de 28 números, por lo que si no se llega a un término
igual al origen o al anterior o a la unidad, en el peor de los casos puede calcular a lo sumo 27 términos (sin contar el origen)

Autor/es: Tomas Aguirre, Menchaca Miguel Angel, Paulo Sosa

Versión: 01

Fecha: 19/10/2021

Análisis de Casos

1- Numero Abundante =12: "NA<Σ divisores"=====> 12<1+2+3+4+6=16

2- Numero Deficiente =14: "ND>Σ divisores"====> 14>1+2+7=10

3- Numero Perfecto =28: "NP=Σ divisores"====> 28=1+2+3+4+7+14=28

4- Numero Casi Perfecto =16: "NCP=Σ divisores +1"====> 16= 1+2+4+8(+1)

5- Numero Ambicicio =95:====> divisores de 95--->1+5+19=25; divisores de 25--->1+5=6 (Numero Perfecto)

6- Numeros Amigos =220 y 284: "NAM1=Σdivisores de NAM2 y NAM2=Σdivisores de NAM1"====> div 220(1+2+4+5+10+20+22+44+55+110)=284

====> div

284(1+2+4+71+142)=220

7- Numeros Casi Amigos =48 y 75: "NAM1=Σdivisores de NAM2-1 y NAM2=Σdivisores de NAM1-1"====> div 48((1+3+5+15+25(-1))=75

====> div

75(1+2+3+4+6+8+12+16+24(-1))=48

8- Numeros Sociables =12496(NAM1), 14288(NAM2), 15472(NAM3), 14536(NAM4) y 14264(NAM5):
"NAM2=Σdivisores de NAM1, NAM3=Σdivisores de NAM2, NAM4=Σdivisores de NAM3, NAM5=Σdivisores de NAM4 y NAM1=Σdivisores de NAM5"

Síntesis de Casos (composición de casos para la generalización):

El programa va a solicitar un numero natural y positivos para asi hallar su sucesion alicuota y clasificar sus numeros
resultantes (Abundante, Deficiente, Perfecto, Amigos, Casi Amigos, Casi Perfecto, Ambicioso y Sociables)

Recursos (variables y funciones del programa -nombre y propósito)

Datos a solicitar al usuario (sea en el prólogo o sea durante la resolución):

Nat = Numero natural = Nat1

Auxiliares (necesarios para transformaciones intermedias):

div = divisores

uno = valor igualado para condicionar el while

div = divisores

Nat2=Nat1

Resultados (a informar sea durante el desarrollo o en el epílogo):

TER = terminos de la sucesion

termino = terminos de la sucesion

suma = acumulador de divisores

sumatoria = suma

Sección Algorítmica

```
print("╔═══════════════════════════════════════════════════════════════════════════════╗
")
print("║        Trabajo Practico: SUCESION ALICUOTA          ║")
print("╚═══════════════════════════════════════════════════════════════════════════════╝
")

print()
#1.1.2 Descripción o aclaraciones al usuario (opcional)
print('Ingrese un numero natural (numero entero mayor a 0)')
print()
print('El programa mostrara su sucesion alicuota y sus caracteristicas')
print()
#1.2 Datos iniciales
#1.2.1 Solicitud e ingreso de datos desde
Nat=Nat1=int(input('Ingrese un numero natural o 0 para terminar: '))
#1.2.2 Establecimiento de valores iniciales para datos auxiliares o que se transformarán en
resultados (opcional)
div=0
suma=0
sumatoria=0
uno=0
termino=0
Nat2=Nat1
TER=0

#2 RESOLUCIÓN

while Nat>0 or Nat1>0:
    if Nat==1 or Nat1==1 and sumatoria!=1:
        uno=0
        print (uno)
        sumatoria=0
```

```
    if uno==0:
        Nat=Nat1=int(input('\nIngrese un numero natural o 0 para terminar: '))

if Nat>1:
    while Nat>1:
        while Nat>div:
            div+=1
            if Nat%div==0 and div!=Nat:
                suma+=div
                sumatoria=suma
                Nat1==Nat
        TER+=1
        if TER>28:
            print('MAS DE 28 TERMINOS ABUNDANTES')
            div=0
            suma=0
            Nat=Nat1
            sumatoria=0
            termino=0
            TER=0

        print(sumatoria,end=',')
        termino+=1

        #PRIMO
        if  Nat1==2 or Nat1==3:
            print('El origen es PRIMO',end=',')
            termino=0
            Nat1=Nat
        elif Nat1%2!=0 and Nat1!=2:
            raiz=int(Nat1**0.5)
            div=3
            while div<=raiz and Nat1%div!=0:
                div+=2
            if div>raiz:
                print('El origen es PRIMO',end=',')
                div=0
                suma=0
                termino=0
                TER=0

        #PERFECTO
        if Nat1==sumatoria and Nat==Nat1:
            print('El origen es PERFECTO',end=',')
            div=0
            suma=0
            Nat=Nat1
            sumatoria=0
```

```
    termino=0
    TER=0


#AMBICIOSO FALTA
if Nat1!=sumatoria and sumatoria==6:
    print('El origen es AMBICIOSO',end=',')
    Nat1=Nat2
    div=0
    suma=0
    Nat=Nat1
    sumatoria=0
    termino=0
    TER=0


#AMIGOS
if Nat1==sumatoria and Nat!=Nat1:
    if termino==2:
        print('Son AMIGOS',end=',')
        div=0
        suma=0
        Nat=Nat1
        sumatoria=0
        termino=0
        TER=0

    #SOCIOS
    elif termino>2:
        print('Son SOCIOS',end=',')
        div=0
        suma=0
        Nat=Nat1
        sumatoria=0
        termino=0
        TER=0

div=0
suma=0
Nat=sumatoria

#PENULTIMO
if sumatoria==1 and Nat1!=2 and Nat1!=3:
    if Nat2%2==0:
        print('sucesión finita con un primo como penúltimo término',end=',')
    else:
        raiz=int(Nat2**0.5)
        div1=3
```

```
            while div1<=raiz and Nat2%div1!=0:
                div1+=2
            if div1>raiz:
                print('')
            else:
                print('sucesión finita con un primo como penúltimo término',end=',')


        div=0
        suma=0
        sumatoria=0
        termino=0
        TER=0


    print()
    Nat=Nat1=int(input('\nIngrese un numero natural o 0 para terminar: '))
    Nat2=Nat1
    if Nat1==1 and sumatoria==1:
        print (uno)
        sumatoria=0
        if uno==0:
            Nat=Nat1=int(input('\nIngrese un numero natural o 0 para terminar: '))
            TER=0


if Nat == 0:
    print()
    print('Gracias por utilizar el programa')

#3 EPÍLOGO
#3.1 Muestra de la solución del problema por pantalla (opcional, si sólo se muestran
resultados durante la resolución)

#3.2 Pausa para ver resultados en pantalla que se puede obviar, si los resultados se van
mostrando durante la resolución
print() # salto de línea
input('Pulse tecla Enter para terminar el programa...') # pausa forzada
```