El Manejo de Números en Código

Por: **David Santiago Sierra, Edgar Duván Bernal Acero y Santiago Naranjo Herrera**

Introducción

En el presente informe se detallan las soluciones implementadas para tres problemas planteados en la materia de Análisis Numérico. Los problemas consistieron en desarrollar códigos en Python para:

- Encontrar todos los números primos hasta un número n dado
- Convertir un número arábico a su equivalente en números romanos
- Representar los números en texto.

A continuación, se describe el enfoque tomado para cada problema, así como los resultados obtenidos y las conclusiones derivadas de las implementaciones. También se realiza un análisis individual sobre la importancia de cada problema y el cómo podría ser implementado en las labores actuales de la sociedad.

Primer Código: Hallar los números Primos Hasta n

Descripción del Problema

El objetivo de realizar el programa fue implementar un algoritmo que calcule todos los números primos hasta un valor n dado por el usuario.

Metodología

Para resolver este problema se realizó un enfoque que fue basado en la definición de "número primo", la cual dice que un número primo es aquel que únicamente tiene solo dos divisores positivos enteros, él mismo y el 1.

Así mismo, debido a que los números compuestos tienen una definición opuesta a la de los números primos, se realizó un análisis que concluyó en que solo los números compuestos pueden factorizarse. Es decir, que todos los números que sean múltiplos de más de un número son considerados números compuestos.

Con esta información, se procede a crear un programa que solicita un límite para comprender la sucesión de números primos. Luego, se crea una tabla de números que comprende un rango de 2 hasta n (Debido a que el primer número primo es el número 2). Por último se procede a recorrer toda la tabla marcando todos los múltiplos del número en cuestión, indicando que dichos números han sido explorados más de una vez y por lo tanto no son primos.

Por último, se imprime una lista que recorre toda la tabla de números e imprime solo aquellos números que no han sido marcados, es decir, todos los números que no fueron múltiplos de otras variables.

Análisis del Problema

Un algoritmo para encontrar números primos puede tener aplicaciones significativas en la seguridad informática, especialmente en el campo de la criptografía. Podemos usar números primos como un factor fundamental dentro de la generación de claves de cifrado en sistemas que requieren contraseñas complejas, debido a que allí se utilizan números primos grandes para crear claves seguras que protegen la información en internet, como las transacciones bancarias, correos electrónicos y otros datos sensibles. A su vez, también podemos ver que la velocidad con la que se hacen estas operaciones hoy en día eran impensables en la antigüedad, dando a entender una creciente evolución en la forma de calcular operaciones matemáticas para resolver problemas en la actualidad.

```
def numerosPrimos():
    n = int(input("Ingresa un número entero: "))
    numeros = dict()
    numeroInicial = 2

    # Listar todos los numeros del 2 hasta n y contarlos como no marcados
    while(numeroInicial<=n):
        numeros[numeroInicial] = 0
        numeroInicial+=1

    for elemento in numeros:
        if numeros[elemento]==0:
            iterador = elemento
            while (iterador<=n/elemento):
                numeros[iterador*elemento] = 1
                iterador+=1
```

```
    print("Los números primos hasta:",n, " Son:")
    for elemento in numeros:
        if numeros[elemento]==0:
            print(elemento, " ")


numerosPrimos()
```

Segundo Código: Conversión de Números Arábicos a Romanos

Descripción del Problema

El segundo problema consistió en implementar un algoritmo que convierta un número arábico (teniendo la condición de ser un número entero positivo) a su representación en números romanos.

Metodología

Para la conversión se utilizó un enfoque basado en la reducción del número y la concatenación de su símbolo equivalente en romano. También se aprovechó la sintaxis de python, que permite multiplicar caracteres y cadenas para ahorrar el proceso de hacer un ciclo repetitivo por cada reducción. La idea principal del algoritmo es reducir el número verificando todos los casos especiales en los que pueden ir dos símbolos en vez de uno, dando a entender que si ningún caso se cumplía, se procedía a concatenar un único símbolo equivalente las veces que fuesen necesarias.

Análisis

Los algoritmos de conversión destacan la capacidad de la programación para adaptar y transformar diferentes sistemas de numeración. Implementar un convertidor de números arábicos a romanos en un programa de computadora nos muestra cómo las matemáticas pueden ser utilizadas no solo en cálculos numéricos, sino también en la manipulación y representación de datos, lo que es crucial en la preservación y transmisión de conocimientos históricos y culturales; permitiendo dar un indicio de que la lógica matemática puede ser utilizada para realizar traducciones entre lenguajes e idiomas.

```
def arabicoaRomano():
    # Solicitar el Número
    numero = int(input("Por Favor Ingresa un número entre 1 y 3000: "))
    if(numero<1 or numero >3000):
        return "El número Ingresado no es válido, por favor inténtalo
nuevamente."

    numeroRomano = ""
    # Reducir los millares del número
    if numero >= 1000:
        m = numero//1000
        numeroRomano+='M'*m
        numero%=1000
    # Reducir las centenas del número
    if numero >= 100:
        c = numero//100
        if c==9:
            numeroRomano+="CM"
        elif c >= 5:
            numeroRomano+='D'+'C'*(c-5)
        elif c==4:
            numeroRomano+="CD"
        else:
            numeroRomano+'C'*c
        numero%=100
    # Reducir las décimas del Número
    if numero >= 10:
        d = numero//10
        if d == 9:
            numeroRomano+="XC"
        elif d >= 5:
            numeroRomano+='L'+'X'*(d-5)
        elif d == 4:
            numeroRomano+='XL'
        else:
            numeroRomano+='X'+d
        numero%=10
    # Reducir el Restante del número
    if numero>0:
        if numero == 9:
            numeroRomano+='IX'
        elif numero >= 5:
            numeroRomano+='V'+'I'*(numero-5)
        elif numero == 4:
            numeroRomano+="IV"
        else:
```

```
            numeroRomano+='I'*numero

    return numeroRomano

print(arabicoaRomano())
```

Tercer Código: Conversión de Número a Texto

Descripción del Problema

El tercer problema planteado fue crear una función que convierta un número arábico en su equivalente en palabras.

Metodología

Se empleó un enfoque basado en la descomposición del número en sus unidades, decenas, centenas, etc., utilizando listas para manejar las palabras correspondientes.

> Debido a que el código es extenso, se optó por publicarlo en forma de texto plano dentro de una página de gist de GitHub que tiene el siguiente link: Código

Análisis

La conversión de números a texto puede aplicarse dentro del campo de muchas utilidades como por ejemplo: el desarrollo de sistemas de voz automatizados, generación de cheques, facturación y en dispositivos de asistencia para personas con discapacidades visuales. Esto nos demuestra que el análisis matemático para problemas que en un principio se puede creer que son triviales, realmente puede ayudar a mejorar la legibilidad de un formato que se usa a diario y que es fundamental para las personas que tienen algún tipo de discapacidad.

Conclusión

El desarrollo de los códigos en Python para los problemas propuestos permitió afianzar los conocimientos sobre algoritmos de manipulación numérica y su implementación en

lenguaje de programación. Los resultados obtenidos fueron satisfactorios, cumpliendo con los objetivos establecidos para cada problema.