ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS
MÉTODOS NUMÉRICOS
INGENIERÍA DE CIENCIAS DE LA COMPUTACIÓN

Nombre: Moisés Pineda

Fecha: 04/05/2025

Curso: GR1CC

Docente: Jonathan A. Zea

Repositorio:
https://github.com/SantiagoTmg/Metodos_Numericos_GRCC1/tree/dec4b794b33264a9f4f8eb7329439228d2ff91bd/Tareas/%5BTarea%2003%5D%20Ejercicios%20Unidad%2001-B

CONJUNTO DE EJERCICIOS 1.3

1. **Utilice aritmética de corte de tres dígitos para calcular las siguientes sumas. Para cada parte, ¿Qué método es más preciso y por qué?**

   a.- $\sum_{i=1}^{10}\left(1/_{i^2}\right)$ primero por $\frac{1}{1}+\frac{1}{4}+\cdots+\frac{1}{100}$ y luego por $\frac{1}{100}+\frac{1}{81}+\cdots+\frac{1}{1}$

   Pseudocódigo:

```
INICIO
  DEFINIR función SUMA_TRUNCADA_SENTIDO_ASCENDENTE(n, digitos, funcion)
    terms ← lista vacía
    suma ← 0
    contador ← 0

    PARA i DESDE 1 HASTA n HACER
      suma ← suma + (1 / i^2)
      contador ← contador + 1

      SI contador = 1 ENTONCES
        suma ← funcion(suma, 3)
        contador ← 0
      FIN SI

      AGREGAR suma A terms
    FIN PARA

    RETORNAR funcion(suma, digitos), terms
  FIN FUNCIÓN


  DEFINIR función SUMA_TRUNCADA_SENTIDO_DESCENDENTE(n, digitos, funcion)
    terms ← lista vacía
    suma ← 0
    contador ← 0
```

```
        PARA i DESDE n HASTA 1 HACER
            suma ← suma + (1 / i^2)
            contador ← contador + 1

            SI contador = 1 ENTONCES
                suma ← funcion(suma, 3)
                contador ← 0
            FIN SI

            AGREGAR suma A terms
        FIN PARA

        RETORNAR funcion(suma, digitos), terms  # Truncamiento final
    FIN FUNCIÓN
FIN
```

Respuestas:
Suma truncada ascendente: 1.53
Suma truncada descendente: 1.54

Conclusión: La suma truncada en sentido descendente muestra mayor precisión, esto se debe a que los términos más pequeños se agregan primeros. Esto minimiza los errores acumulados en la aritmética de corte de tres dígitos.

b.- $\sum_{i=1}^{10}\left({}^{1}/{}_{i^3}\right)$ primero por $\frac{1}{1}+\frac{1}{8}+\frac{1}{27}+\cdots+\frac{1}{1000}$ y luego por $\frac{1}{1000}+\frac{1}{729}+\cdots+\frac{1}{1}$

Pseudocódigo:

```
INICIO
    DEFINIR función SUMA_TRUNCADA_SENTIDO_ASCENDENTE(n, digitos, funcion)
        terms ← lista vacía
        suma ← 0
        contador ← 0

        PARA i DESDE 1 HASTA n HACER
            suma ← suma + (1 / i^3)
            contador ← contador + 1

            SI contador = 1 ENTONCES
                suma ← funcion(suma, 3)
                contador ← 0
            FIN SI

            AGREGAR suma A terms
```

```
    FIN PARA

    RETORNAR funcion(suma, digitos), terms
  FIN FUNCIÓN

  DEFINIR función SUMA_TRUNCADA_SENTIDO_DESCENDENTE(n, digitos, funcion)
    terms ← lista vacía
    suma ← 0
    contador ← 0

    PARA i DESDE n HASTA 1 HACER
      suma ← suma + (1 / i^3)
      contador ← contador + 1

      SI contador = 1 ENTONCES
        suma ← funcion(suma, 3)
        contador ← 0
      FIN SI

      AGREGAR suma A terms
    FIN PARA

    RETORNAR funcion(suma, digitos), terms  # Truncamiento final
  FIN FUNCIÓN
FIN
```

Respuestas:
Suma truncada ascendente: 1.15
Suma truncada descendente: 1.19

Conclusión: La suma truncada en sentido descendente nuevamente presenta una mayor precisión. Esto se explico en el anterior literal.

2. **La serie de Maclaurin para la función arcotangente converge para $-1 < x \leq 1$ y está dada por**

$$\arctan x = \lim_{n\to\infty} P_n(x) = \lim_{n\to\infty} \sum_{i=1}^{n} (-1)^{i+1} \frac{x^{2i-1}}{2i-1}$$

a.- Utilice el hecho de que $\tan {}^{\pi}/_{4} = 1$ para determinar el número n de términos de la serie que se necesita sumar para garantizar que $|4P_n(1) - \pi| < 10^{-3}$.

<u>Se tiene que</u> $\tan\left(\frac{\pi}{4}\right) = 1$, <u>entonces</u>: $\arctan(1) = \frac{\pi}{4}$

$$\arctan(1) = \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1}$$

$$\frac{\pi}{4} = \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1}$$

$$P_n(1) = \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1}$$

$$|4P_n(1) - \pi| < 10^{-3}$$

$$|4 \cdot \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1} - \pi| < 10^{-3}$$

Pseudocódigo:

```
INICIO
    DEFINIR función ERROR_ABSOLUTO(valor)
        RETORNAR ABS(4 * valor - π)
    FIN FUNCIÓN

    DEFINIR función EXPRESION(n)
        RETORNAR ((-1)^(n+1)) / (2*n - 1)
    FIN FUNCIÓN

    DEFINIR función SUMATORIA(error_maximo)
        sumatoria ← 0
        contador ← 0

        MIENTRAS ERROR_ABSOLUTO(sumatoria) > error_maximo HACER
            contador ← contador + 1
            sumatoria ← sumatoria + EXPRESION(contador)
        FIN MIENTRAS

        RETORNAR sumatoria, contador
    FIN FUNCIÓN
FIN
```

Respuesta:
Número de términos necesarios: 1000
Valor aproximado de π: 3.140592653839794

b.- El lenguaje de programación C++ requiere que el valor de π se encuentre dentro de 10^{-10}. ¿Cuántos términos de la serie se necesitarían sumar para obtener este grado de precisión?

Pseudocódigo:

```
INICIO
```

```
  DEFINIR FUNCIÓN SUMATORIA_CON_LIMITACION(error_maximo)
    sumatoria ← 0
    contador ← 1
    mensaje ← ""

    MIENTRAS VERDADERO HACER  # Ejecutar hasta alcanzar el error deseado
      sumatoria ← sumatoria + EXPRESION(contador)

      SI ERROR_ABSOLUTO(sumatoria) < error_maximo ENTONCES
        TERMINAR BUCLE
      FIN SI

      contador ← contador + 1

      SI contador ≠ 1000000 ENTONCES
        mensaje ← "Número de términos necesarios: " + contador
      SINO
        mensaje ← "Número de iteraciones máximo alcanzado: " + contador
        TERMINAR BUCLE
      FIN SI
    FIN MIENTRAS

    RETORNAR sumatoria, mensaje
  FIN FUNCIÓN
FIN
```

Respuesta:
Número de iteraciones máximo alcanzado: 1000000
Valor aproximado de π: 3.1415936535907742

Conclusión: Debido a que se necesitan evaluar una gran cantidad de términos, no se pudo llegar a la aproximación requerida.

3. **Otra fórmula para calcular π se puede deducir a partir de la identidad $\pi/4 = 4\tan^{-1}\frac{1}{5} - \tan^{-1}\frac{1}{239}$. Determine el número de términos que se deben sumar para garantizar una aproximación π dentro de 10^{-3}.**

SE TIENE QUE: $\arctan(x) = \sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{2n+1}$

Entonces,

$$\pi_{calculado} \approx 4(4\sum_{n=0}^{\infty}\frac{(-1)^n(\frac{1}{5})^{2n+1}}{2n+1} - \sum_{n=0}^{\infty}\frac{(-1)^n(\frac{1}{239})^{2n+1}}{2n+1}$$

$$|\pi_{calculado} - \pi| < 10^{-3}$$

Pseudocódigo:

```
INICIO
  DEFINIR FUNCIÓN ARCTAN(x, n)
    suma ← 0

    PARA i DESDE 0 HASTA n-1 HACER
      expresion ← ((-1)^i) * (x^(2*i+1)) / (2*i + 1)
      suma ← suma + expresion
    FIN PARA

    RETORNAR suma
  FIN FUNCIÓN

  DEFINIR FUNCIÓN CALCULO_PI(error_maximo)
    n ← 1
    pi_aproximado ← 4 * (4 * ARCTAN(1/5, n) - ARCTAN(1/239, n))

    MIENTRAS ABS(pi_aproximado - π) > error_maximo HACER
      n ← n + 1
      pi_aproximado ← 4 * (4 * ARCTAN(1/5, n) - ARCTAN(1/239, n))
    FIN MIENTRAS

    RETORNAR n, pi_aproximado
  FIN FUNCIÓN
FIN
```

Respuesta:
Número de términos necesarios: 2
Valor de pi calculado: 3.1405970293260603
Error absoluto: 0.0009956242637327861

4. **Compare los siguientes tres algoritmos. ¿Cuándo es correcto el algoritmo de la parte 1a?**
   a) ENTRADA $n, x_1, x_2, \cdots, x_n$.
      SALIDA PRODUCT
      Paso 1 Determine PRODUCT = 0.
      Paso 2 Para $i = 1, 2, \cdots, n$ haga
         Determine PRODUCT = PRODUCT $* \, x_i$.
      Paso 3 SALIDA PRODUCT;
         PARE
   b) ENTRADA $n, x_1, x_2, \cdots, x_n$.
      SALIDA PRODUCT
      Paso 1 Determine PRODUCT = 1.
      Paso 2 Para $i = 1, 2, \cdots, n$ haga

Set PRODUCT = PRODUCT $* x_i$.

Paso 3 SALIDA PRODUCT;
PARE

c) ENTRADA $n, x_1, x_2, \cdots, x_n$.
SALIDA PRODUCT
Paso 1 Determine PRODUCT = 1.
Paso 2 Para $i = 1, 2, \cdots, n$ haga
si $x_i = 0$ entonces determine PRODUCT = 0;
SALIDA PRODUCT;
PARE
Determine PRODUCT = PRODUCT $* x_i$.
Paso 3 SALIDA PRODUCT;
PARE.

Respuesta: El algoritmo a) solo es correcto cuando todos los elementos de entrada son cero. Esto se debe a que al iniciar a producto se le asigno el valor de cero y cualquier número multiplicado para este da cero. El algoritmo b) es correcto ya que le asigna el valor de 1 a producto. Por último, el algoritmo c) es más optimo ya que si detecta una entrada de cero directamente retorna cero, por esa razón, lo hace mas optimo.

5. a.- ¿Cuántas multiplicaciones y sumas se requieren para determinar una suma de la forma $\sum_{i=1}^{n} \sum_{j=1}^{i} a_i b_j$?

Teniendo en cuenta la siguiente expresión:

$$\sum_{i=1}^{n} \sum_{j=1}^{i} a_i b_j$$

Esto indica que para cada valor de i, se suman los productos de $a_i b_j$ desde $j = 1$ hasta $j = i$. Esto significa que la expresión representa una suma en la que, para cada i, el elemento a_i se multiplica por cada uno de los b_j hasta ese valor de i.

Entonces, la expresión completa equivale a:

$$a_1 b_1 + a_2 b_1 + a_2 b_2 + a_3 b_1 + a_3 b_2 + a_3 b_3 + \cdots + a_n b_1 + \cdots + a_n b_n$$

Esto quiere decir que el resultado será una suma acumulativa de productos cruzados entre los vectores a y b, donde el índice del segundo sumado (el de b_j) se extiende hasta el valor actual de i.

Multiplicaciones
Para cada i, se realizan i productos $a_i b_j$. La suma total de multiplicaciones será:

$$1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$$

Sumas
La cantidad de sumas es igual a la cantidad de términos menos 1:

$$\frac{n(n+1)}{2} - 1$$

Pseudocódigo:

```
INICIO
  DEFINIR FUNCIÓN CONTAR_OPERACIONES(n)
    multiplicaciones ← 0
    sumas ← 0

    PARA i DESDE 1 HASTA n HACER
      PARA j DESDE 1 HASTA i HACER
        multiplicaciones ← multiplicaciones + 1
        SI i ≠ 1 O j ≠ 1 ENTONCES
          sumas ← sumas + 1
        FIN SI
      FIN PARA
    FIN PARA

    RETORNAR multiplicaciones, sumas
  FIN FUNCIÓN
FIN
```

Respuesta:
Para n = 6:
Total de multiplicaciones: 21
Total de sumas: 20

b.- Modifique la suma en la parte a) a un formato equivalente que reduzca el número de cálculos.

La suma doble original:

$$\sum_{i=1}^{n}\sum_{j=1}^{i} a_i b_j$$

Mediante la propiedad de conmutativa de la suma doble, se ha intercambiado el orden de los sumatorios y se logró identificar un patrón que permite factorizar y reducir el número de operaciones.

En resumen, en vez de repetir muchas veces el producto $a_i b_j$, se agrupa la cantidad de veces que cada par $a_i b_j$ aparece, lo cual ocurre exactamente $(n - j + 1)$ veces, para todos los $i \geq j$.

De este modo, cada termino de la forma $a_i b_j$ se multiplica directamente por el numero de veces que habría aparecido en la suma original.

La expresión optimizada se puede expandir de la siguiente manera:

$$a_i b_j(n) + a_2 b_2(n-1) + a_3 b_3(n-2) + \cdots + a_n b_n(1)$$

Multiplicaciones

Cada termino de la forma $a_j b_j(n-j+1)$ requiere 2 multiplicaciones: una para $a_j \cdot b_j$ y otra para multiplicar ese resultado por $(n-j+1)$. Por lo tanto:

$$\boldsymbol{Total\ de\ multiplicaciones} = 2n$$

Sumas

Se realizan $n-1$ sumas para combinar los n términos:

$$\boldsymbol{Total\ de\ sumas} = n-1$$

Pseudocódigo:

```
INICIO
   DEFINIR FUNCIÓN CONTAR_OPERACIONES_MODIFICADA(n)
      multiplicaciones ← 2 * n
      sumas ← n - 1

      RETORNAR multiplicaciones, sumas
   FIN FUNCIÓN
FIN
```

Respuesta:

Para n = 6:

Total de multiplicaciones: 12

Total de sumas: 5

DISCUSIONES

1. **Escriba un algoritmo para sumar la serie finita $\sum_{i=1}^{n} x_i$ en orden inverso.**

Pseudocódigo:

```
INICIO
   DEFINIR FUNCIÓN SUMA_FINITA(lista)
      suma ← 0

      PARA i EN lista EN ORDEN INVERSO HACER
         suma ← suma + i
      FIN PARA

      RETORNAR suma
   FIN FUNCIÓN
FIN
```

2. Las ecuaciones (1.2) y (1.3) en la sección 1.2 proporcionan formas alternativas para las raíces x_1 y x_2 de $ax^2 + bx + c = 0$. Construya un algoritmo con entrada a, b, c c y salida x_1, x_2 que calcule las raíces $x_1\, y\, x_2$ (que pueden ser iguales con conjugados complejos) mediante la mejor fórmula para cada raíz.

Pseudocódigo:

```
INICIO
   DEFINIR FUNCIÓN CALCULO_DE_RAICES(a, b, c)
      discriminante ← b^2 - 4*a*c

      SI discriminante > 0 ENTONCES
         SI b > 0 ENTONCES
            x1 ← (-b - sqrt(discriminante)) / (2*a)
            x2 ← (2*c) / (-b - sqrt(discriminante))
         SINO
            x1 ← (-b + sqrt(discriminante)) / (2*a)
            x2 ← (2*c) / (-b + sqrt(discriminante))
         FIN SI
         RETORNAR "Raíces reales: x1 =", x1, "x2 =", x2

      SI discriminante = 0 ENTONCES
         x ← -b / (2*a)
         RETORNAR "Raíces iguales y reales: x =", x

      SINO
         p_real ← -b / (2*a)
         p_imaginaria ← sqrt(ABS(discriminante)) / (2*a)
         RETORNAR "Raíces complejas: x1 =", p_real, "+", p_imaginaria, "i,", "x2 =",
p_real, "-", p_imaginaria, "i"
   FIN FUNCIÓN
FIN
```

3. Suponga que

$$\frac{1-2x}{1-x+x^2} + \frac{2x-4x^3}{1-x^2+x^4} + \frac{4x^3-8x^7}{1-x^4+x^8} + \cdots = \frac{1+2x}{1+x+x^2},$$

Para $x < 1$ y si $x = 0.25$. Escriba y ejecute un algoritmo que determine el número de términos necesarios en el lado izquierdo de la ecuación de tal forma que el lado izquierdo difiera del lado derecho en menos de 10^{-6}.

Se tiene que:

$$\frac{1-2x}{1-x+x^2} + \frac{2x-4x^3}{1-x^2+x^4} + \frac{4x^3-8x^7}{1-x^4+x^8} + \cdots$$

Lo cual es lo mismo que:

ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS
MÉTODOS NUMÉRICOS
INGENIERÍA DE CIENCIAS DE LA COMPUTACIÓN

$$\sum_{n=1}^{\infty} \frac{2^{n-1}x^{2^{n-1}-1} - 2^n x^{2^n-1}}{1 - x^{2^{n-1}} + x^{2^n}}$$

Pseudocódigo:

```
INICIO
  DEFINIR FUNCIÓN EXPRESION_IZQUIERDA(n, x)
    RETORNAR (2^(n-1) * x^(2^(n-1)-1) - 2^n * x^(2^n-1)) / (1 - x^(2^(n-1)) + x^(2^n))
  FIN FUNCIÓN

  DEFINIR FUNCIÓN EXPRESION_DERECHA(x)
    RETORNAR (1 + 2*x) / (1 + x + x^2)
  FIN FUNCIÓN

  DEFINIR FUNCIÓN SUMATORIA3(n, x)
    suma ← 0

    PARA i DESDE 1 HASTA n HACER
      suma ← suma + EXPRESION_IZQUIERDA(i, x)
    FIN PARA

    RETORNAR suma
  FIN FUNCIÓN

  DEFINIR FUNCIÓN CALCULO_TERMINOS(x, errorLimite)
    n ← 1

    MIENTRAS ABS(SUMATORIA3(n, x) - EXPRESION_DERECHA(x)) > errorLimite HACER
      n ← n + 1
    FIN MIENTRAS

    RETORNAR n
  FIN FUNCIÓN
FIN
```

Respuesta:
Número de términos: 4
Aproximación: 1.1428571279559818