Deber Metodos Numericos Parte 1

Jose Armando Sarango Cuenca

Tabla de Contenidos

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). Σ(n=1 → 10) (1/n^2)‘ 1/1+1/2+1/4+...+1/100 y luego por 1/100+1/81+...+1/1

```
def suma1():
    suma = 0
    for i in range(1, 11):
        suma += 1 / (i ** 2)
    #return suma
    return round(suma,3)

def suma2():
    suma = 0
    for i in range(1, 11):
        suma += round(1 / ((11 - i) ** 2),3)
    return round(suma,3)

resultado1 = suma1()
resultado2 = suma2()

print("Resultado de la primera suma:", resultado1)
print("Resultado de la segunda suma:", resultado2)
```

```
Resultado de la primera suma: 1.55
Resultado de la segunda suma: 1.549
```

b) Σ(n=1 → 10) (1/n^3)‘ 1/1+1/8+1/27+...+1/1000 y luego por 1/1000+1/729+...+1/1

```python
def suma1():
    suma = 0
    for i in range(1, 11):
        suma += 1 / (i ** 3)
    #return suma
    return round(suma,3)

def suma2():
    suma = 0
    for i in range(1, 11):
        suma += round(1 / ((11 - i) ** 3),3)
    return round(suma,3)

resultado1 = suma1()
resultado2 = suma2()

print("Resultado de la primera suma:", resultado1)
print("Resultado de la segunda suma:", resultado2)
```

```
Resultado de la primera suma: 1.198
Resultado de la segunda suma: 1.198
```

2) La serie de Maclaurin para la función arcotangente converge para -1<x 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 /4 = 1 para determinar el número n de términos de la serie que se necesita sumar para garantizar que |4P__n (1-)|<10^(-3) |

```python
import math

def maclaurin_arctan_terms(epsilon):
    n = 1
    approx_pi = 0
    term = 1  # Initial term for x = 1
    x = 1

    while abs(4 * approx_pi - math.pi) >= epsilon:
        approx_pi += (-1)**(n + 1) * x**(2 * n - 1) / (2 * n - 1)
        n += 1
```

```
    return n - 1  # Subtract 1 because n was incremented one extra time

n_terms = maclaurin_arctan_terms(1e-3)
print(f"Número de términos para |4Pn(1) -  | < 10^-3: {n_terms}")
n_terms = maclaurin_arctan_terms(1e-10)
```

```
Número de términos para |4Pn(1) -  | < 10^-3: 1000
```

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

```
print(f"Numero de terminos para |4Pn(1) -  | < 10^-10: {n_terms}")
```

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

```
def calculate_pi_epsilon(epsilon):
    n = 1
    pi_approx = 0
    term_1 = 1 / 5
    term_2 = 1 / 239

    while abs(pi_approx - math.pi) >= epsilon:
        pi_approx = 4 * (4 * sum_series_maclaurin(1 / 5, n) - sum_series_maclaurin(1 / 239, n
        n += 1

    return n - 1

def sum_series_maclaurin(x, terms):
    total_sum = 0
    for i in range(1, terms + 1):
        total_sum += (-1)**(i + 1) * x**(2 * i - 1) / (2 * i - 1)
    return total_sum

n_terms = calculate_pi_epsilon(1e-3)
print(f"Número de términos para la aproximación   dentro 10^-3: {n_terms}")
```