RECURSO PROGRAMACIÓN FUNCIÓN SENO

La función seno sin(x) es una de las funciones trigonométricas más utilizadas en matemáticas, física e ingeniería. Aunque los lenguajes de programación tienen funciones nativas para calcularla, es interesante aprender cómo se puede programar desde cero usando la serie de Taylor. Este método nos enseña los fundamentos del cálculo numérico, las aproximaciones y la programación iterativa.

El desarrollo de la serie de Taylor para calcular el seno de un ángulo x es:

$$\text{Sen(x)} = \sum_{k=0}^{n} \frac{(-1)^{K} * x^{(2K+1)}}{(2k+1)!}$$

Implementar la serie de Taylor permite entender cómo las computadoras aproximan funciones trascendentales como el seno a través de una suma finita de términos.

Explicación fórmula:
La fórmula consiste en la serie de Taylor para aproximar el seno de un ángulo x. Esta serie representa sin(x) como una suma infinita de términos. A continuación, se explica cada elemento de la fórmula:

1. El símbolo de suma $\sum_{k=0}^{n}$
○ Indica que la fórmula es una suma acumulativa de términos.
○ k es el índice que controla cada término, empezando en 0 y terminando en n, el número de términos.

2. (-1)^k
○ Alterna el signo de los términos. Para k=0,1,2,… los signos serán +,−,+,−,+, - y así sucesivamente.
○ Refleja la naturaleza oscilante del seno (sube y baja).

3.$x^{(2K+1)}$
○ Eleva el ángulo x a potencias impares (x^1, x^3, x^5,...)
○ Representa los términos de la expansión de la función seno.

4. (2k+1)!(2k+1)!(2k+1)!:
○ Es el factorial del exponente impar (1!,3!,5!,…)
○ Sirve como denominador, reduciendo la magnitud de los términos a medida que k aumenta.

5. n:
○ Es el número de términos que decides incluir en la suma. Cuantos más términos uses, más precisa será la aproximación del seno.
Es importante saber que la precisión depende del número de n usados. A mayor n más términos del polinomio de Taylor y mayor precisión.

PROGRAMACIÓN EN R (con anotaciones y explicación)

Programa 1

```
# Inicializar la variable que acumulará el resultado del seno
sinx <- 0  # Inicialmente el valor acumulado del seno es 0

# Calcular el seno mediante la serie de Taylor
for (k in 0:n) {
  # Cada término de la serie se calcula como:
  # Término = [((-1)^k) * x^(2k + 1)] / (2k + 1)!

  # Numerador: Alterna el signo y eleva el ángulo a la potencia impar
  a <- ((-1)^k) * (x^(2 * k + 1))

  # Denominador: Factorial del exponente impar
  b <- factorial(2 * k + 1)

  # Sumar el término actual al acumulador
  sinx <- sinx + a / b
}

# Imprimir los resultados
# Mostrar el valor aproximado calculado con la serie de Taylor
print(paste("El valor aproximado de sin(x) usando la serie de Taylor es:", sinx))

# Comparar con el valor exacto calculado usando la función nativa sin(x)
print(paste("El valor exacto de sin(x) usando la función sin() es:", sin(x)))
```

El primer programa es más didáctico para aprender sobre programación y trigonometría, ya que incluye la conversión explícita de grados a radianes y muestra los pasos necesarios para este proceso.
El segundo programa es más compacto y flexible, permitiendo que el usuario ingrese directamente valores en radianes o incluso expresiones matemáticas, lo que lo hace más versátil pero menos detallado en términos educativos.

Segundo programa

```
cat("\014")          # Limpia la consola para un mejor enfoque en los resultados
rm(list = ls())      # Elimina todos los objetos del entorno de trabajo

# Solicitar el valor del ángulo en radianes
x <- readline(prompt = "Introduce el valor de x (en radianes) para calcular el seno: ")
# La entrada se toma como texto (string)
x <- eval(parse(text = x))
# Convierte la entrada de texto en un valor numérico, permitiendo operaciones matemáticas

# Solicitar el número de términos para la aproximación
n <- as.integer(readline(prompt = "Introduce el número de términos a calcular en la serie de
Taylor: "))

# Inicializar la variable para acumular el resultado del seno
sinx <- 0  # Inicialmente el seno aproximado es 0

# Cálculo del seno usando la serie de Taylor
for (k in 0:n) {
  # Cada término de la serie:
  # Término = [((-1)^k) * x^(2k + 1)] / (2k + 1)!

  # Numerador: Alterna el signo y eleva x a potencias impares
  a <- ((-1)^k) * (x^(2 * k + 1))

  # Denominador: Factorial del exponente impar
  b <- factorial(2 * k + 1)

  # Suma el término actual a la acumulación
  sinx <- sinx + a / b
}

# Mostrar los resultados
# Imprimir el valor aproximado usando la serie de Taylor
print(paste("El valor aproximado de sin(x) usando la serie de Taylor es:", sinx))

# Comparar con el valor exacto usando la función sin()
print(paste("El valor exacto de sin(x) usando la función sin() es:", sin(x)))
```

Ejercicio adicional

Modifica cualquiera de los programas para:

1. Comparar los resultados de la serie con diferentes números de términos (n) y observar cómo mejora la precisión.

```
cat("\014")          # Limpia la consola
rm(list = ls())      # Elimina objetos del entorno

# Solicitar el ángulo en radianes (entrada como texto)
x <- readline(prompt = "Introduce el valor de x (en radianes): ")
x <- eval(parse(text = x))  # Convierte el texto en un valor numérico

# Solicitar el número de términos de la serie
max_n <- as.integer(readline(prompt = "Introduce el número máximo de términos para la
serie de Taylor: "))

# Inicializar una lista para guardar los resultados de la aproximación
resultados <- data.frame(n=integer(), sinx=numeric(), error=numeric())

# Calcular el seno usando la serie de Taylor para diferentes valores de n
for (n in 1:max_n) {
   sinx <- 0  # Inicializar sinx
   for (k in 0:n) {
      a <- ((-1)^k) * (x^(2 * k + 1))  # Numerador
      b <- factorial(2 * k + 1)  # Denominador
      sinx <- sinx + a / b  # Acumular el valor del seno
   }
   # Calcular el error como la diferencia entre el valor exacto y el aproximado
   error <- abs(sinx - sin(x))
   resultados <- rbind(resultados, data.frame(n=n, sinx=sinx, error=error))
}
print("Resultados de la aproximación del seno:")
print(resultados)
```