Práctica de Recursividad de Pila con Números en C#

Escuela de Ingeniería en Computadores
Instituto Tecnológico de Costa Rica

Algoritmos y Estructuras de Datos I

Profesores: M.Sc. Jeff Schmidt Peralta y M.Sc. Jason Leitón Jiménez y M.Sc. Leonardo Araya Martinez
Estudiante: Steven Aguilar Alvarez

Introducción

Este documento presenta una serie de ejercicios diseñados para practicar la manipulación recursiva de números en C#. Los problemas se centran en operaciones matemáticas y análisis de dígitos utilizando recursividad de pila, permitiendo desarrollar habilidades fundamentales en el manejo de números y operaciones recursivas.

1. Ejercicios de Programación

1. **Suma de Dígitos Impares**
   Desarrolle un método recursivo SumOddDigits que reciba un número entero largo y sume todos los dígitos que sean impares. Por ejemplo:
   - SumOddDigits(482401) → 1 (suma del 1)
   - SumOddDigits(3579) → 24 (suma de 3+5+7+9)

2. **Conteo de Dígitos Pares**
   Implemente un método recursivo CountEvenDigits que cuente la cantidad de dígitos pares en un número. Por ejemplo:
   - CountEvenDigits(482401) → 5 (cuenta de 4,8,2,4,0)
   - CountEvenDigits(3579) → 0

3. **Comparación de Dígitos**
   Desarrolle un método recursivo IsFirstLastEqual que verifique si el primer y último dígito de un número son iguales:
   - IsFirstLastEqual(80642) → false
   - IsFirstLastEqual(351733) → true

4. **Suma de Dígitos Mayor a Diez**
   Implemente un método recursivo IsSumAtLeastTen que verifique si la suma de todos los dígitos es mayor o igual a 10:
   - IsSumAtLeastTen(80642) → true (8+0+6+4+2 = 20)
   - IsSumAtLeastTen(200412) → false (2+0+0+4+1+2 = 9)

5. **Ocurrencias de un Dígito**
   Desarrolle un método recursivo CountDigitOccurrences que cuente cuántas veces aparece un dígito específico en un número:
   - CountDigitOccurrences(93235, 3) → 2

6. **Análisis de Dígitos**
   Implemente un método recursivo AnalyzeDigits que retorne una tupla con la cantidad de dígitos entre 0-4 y entre 5-9:
   - AnalyzeDigits(482401) → (5, 1) (5 dígitos entre 0-4, 1 dígito entre 5-9)
   - AnalyzeDigits(4) → (1, 0)
7. **Verificación de Dígitos Pares**
   Desarrolle un método recursivo AreAllDigitsEven que verifique si todos los dígitos de un número son pares:
   - AreAllDigitsEven(80642) → true
   - AreAllDigitsEven(201462) → false
8. **Existencia de Dígitos Pares**
   Implemente un método recursivo HasAtLeastOneEven que verifique si existe al menos un dígito par en el número:
   - HasAtLeastOneEven(80642) → true
   - HasAtLeastOneEven(351733) → false

Observaciones Finales

Los ejercicios deben implementarse utilizando exclusivamente recursividad de pila en C. Es fundamental mantener un control adecuado de la pila de ejecución mediante el uso apropiado de casos base y llamadas recursivas. La implementación debe seguir las convenciones de nomenclatura de C y evitar el uso de estructuras iterativas. Los métodos auxiliares deben utilizarse cuando sea necesario para mantener la claridad y modularidad del código.