UNIDAD TEMÁTICA 2: DISEÑO Y ANÁLISIS DE ALGORITMOS

PRACTICOS DOMICILIARIOS INDIVIDUALES 2

Ejercicio #1

1. Desarrolla en seudocódigo, en forma recursiva, un algoritmo para calcular el factorial de un cierto número entero que se pasa como parámetro.
   - Identifica claramente el caso base y la sentencia que lo contempla.
   - ¿Puedes verificar que siempre el algoritmo progresará hacia el caso base?

2. Analiza el orden del tiempo de ejecución del algoritmo.

3. Implementa el algoritmo (en JAVA) y pruébalo:
   - ¿Qué sucede si el número es negativo?
   - Verifica que factorial(4), factorial(5), y factorial(0) produzcan los resultados

Ejercicio #2

1. Desarrolla en seudocódigo, en forma recursiva, el algoritmo *Algoritmo SumaLineal(A, n),* que se describe en la ppt de clase sobre recursividad.
   - Identifica claramente el caso base y la sentencia que lo contempla.
   - ¿Puedes verificar que siempre el algoritmo progresará hacia el caso base?

2. Analiza el orden del tiempo de ejecución del algoritmo

3. Implementa el algoritmo (en JAVA) y pruébalo:
   - ¿Qué sucede si el parámetro n es negativo?
   - ¿Qué sucede si el vector A está vacío?

Ejercicio #3

1. Desarrolla en seudocódigo, en forma recursiva, un algoritmo para calcular la potencia de un número. El mismo ha de recibir como parámetros el número y el exponente (ver la ppt de clase sobre recursividad).
   - Identifica claramente el caso base y la sentencia que lo contempla.
   - ¿Puedes verificar que siempre el algoritmo progresará hacia el caso base?

2. Analiza el orden del tiempo de ejecución del algoritmo.

3. Implementa el algoritmo (en JAVA) y pruébalo:
   - ¿Tu algoritmo soporta números reales o sólo enteros – para ambos parámetros?
   - ¿qué sucede si uno, otro o ambos parámetros son negativos?

Ejercicio #4

1. Desarrolla en seudocódigo, en forma recursiva, un algoritmo para invertir los componentes de un vector pasado por parámetro, entre dos índices indicados también pasados como parámetros. (ver la ppt de clase sobre recursividad).
   - Identifica claramente el / los caso(s) base y la(s) sentencia(s) que lo contempla(n).
   - ¿Puedes verificar que siempre el algoritmo progresará hacia el caso base?

2. Analiza el orden del tiempo de ejecución del algoritmo.

3. Implementa el algoritmo (en JAVA) y pruébalo:
   - Crea un pequeño vector y prueba el algoritmo. Prueba situaciones de borde (extremos), parámetros fuera de rango, vector vacío, vector con sólo un elemento, etc.