Práctico 2 Complemento

Análisis de algoritmos

EJERCICIO 1 (R)

Dado el siguiente algoritmo se desea calcular la cantidad de operaciones básicas que se realizan en el peor caso.

```
bool funcion (int* a, int i, int j){
      int medio;
      if (i == j)
            return calcular(a, i, i);
      else {
            if (i < j){
                  medio = (i+j-1)/2;
                  if(funcion(a, i, medio))
                        return calcular(a, i, medio);
                  else
                        if(funcion(a, medio + 1, j))
                              return calcular(a, medio + 1, j);
                        else return false;
            }
      }
}
```

La función calcular tiene el siguiente encabezado: *bool calcular(int* a, int i, int j)*

Ambas funciones reciben como parámetro de entrada un arreglo *a* y dos índices *i* y *j* que representan que en esa invocación se consideran los elementos ubicados entre los lugares *i* y *j* del arreglo *a*.

Inicialmente se invoca a *funcion* con los siguientes parámetros: *funcion(a, 1, n)*

La función *calcular* es quién realiza las operaciones básicas, y la cantidad de dichas operaciones es 2(j-i) + 1.

EJERCICIO 2 (R)

Dada la función: f(x) = $\sum_{i=0}^{n} a_i x^i$

Analice el orden del tiempo de ejecución y compare los órdenes en los siguientes casos (considere *n* potencia de 2):

- Utilizando una rutina simple para realizar la exponenciación.
- Utilizando la siguiente rutina

```
int potencia (int x, int n){
  int resultado;

  if (n == 0)
     resultado = 1;
  else if (n == 1)
     resultado = x;
  else if (n % 2 == 0)
     resultado = potencia (x * x, n / 2);
  else
     resultado = potencia (x * x, n / 2) * x;
  return (resultado);
}
```