UNIDAD TEMÁTICA 3: Listas, Pilas, Colas, Orden del Tiempo de ejecución

PRACTICOS DOMICILIARIOS INDIVIDUALES #5– Cálculo del Orden del Tiempo de Ejecución

REFERENCIA: material publicado sobre Análisis del Tiempo de Ejecución de Algoritmos

Ejercicio #1

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
public static int enRango (int[] a, int bajo, int alto) {
      int contador = 0;
      for (int i=0; i<a.length; i++) {
            if (a[i] >= bajo && a[i] < alto)
                  contador++;
      }
      return contador;
}
```

Ejercicio #2

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
unaFunción ( N de tipo entero)
  i ← 1
  j ← N
  mientras i < N hacer
    j ← N – 1
    i ← i * 2
  fin mientras
  devolver (j)
fin
```

Ejercicio #3

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
int[] cuentas = new int [100];
for (int i = 0; i<100; i++) {
  cuentas[i] = enRango (notas, i, i+1);
}
```

Ejercicio #4

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
unValor (A, N de tipos enteros)
i ← 0
Si N < 3 entonces
devolver (A)
fin si
mientras i < 3 hacer
      si arreglo[i] = A entonces
            devolver ((arreglo[0] + arreglo[N-1]) div 2)
      fin si
      i ← i + 1
fin mientras
devolver (A div N)
Fin
```

Ejercicio #5

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
otraFunción (claveAbuscar)
  inicio ← 0
  fin ← N-1
  mientras inicio ≤ fin hacer
    medio ← (inicio + fin) div 2
    si (arreglo[medio] < claveAbuscar) entonces
      inicio ← medio + 1
    sino
      si (arreglo[medio] > claveAbuscar) entonces
        fin ← medio – 1
      sino
        devolver medio
      fin si
    fin si
  fin mientras
  devolver -1
fin
```

Ejercicio #6

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
function particion( i, j: integer; pivote: TipoClave): integer;
{divide V[i], .., V[j] para que las claves menores que pivote estén a la izquierda y las mayores o iguales a la derecha. Devuelve el lugar donde se inicia el grupo de la derecha.}
COMIENZO
  L ← i;
  R ← j;
  Repetir
    intercambia(V[L],V[R]);
    mientras V[L].clave < pivote hacer   L := L + 1; fin mientras
    mientras V[R].clave >= pivote hacer R := R - 1; fin mientras
  Hasta que L > R
  Devolver L;
FIN; {particion}
```

Ejercicio #7

Analiza el orden del tiempo de ejecución del siguiente algoritmo:

```
miFunción
  Desde i = 1 hasta N-1 hacer
    Desde j = N hasta i+1 hacer
      Si arreglo[j].clave < arreglo[j-1].clave entonces
        Intercambia(arreglo[j], arreglo[j-1])
      Fin si
    Fin desde
  Fin desde
Fin
```