PRIMERA EVALUACIÓN
ANÁLISIS DE ALGORITMOS
2018-I

NOMBRE: _____

Tema Uno (10 puntos)

Encuentre la relación de recurrencia y encuentre el límite asintótico superior para el tiempo de ejecución del siguiente algoritmo :

```
void doSomething(arreglo a, int left, int right)
     {
       if (left == right)
          {
          for (int j = 0, j < right)
               {
               print a[j];
               }
          }
       if (a[left]<a[right])
          {
          swap(a[left], a[right]);
          doSomething(a, left, (left+right)/2);
          }
       else
          {
          swap(a[left], a[right]);
          doSomething(a,(left+right)/2, right);
          }
     }
```

Resolución de Tema Uno:

Tema Dos (15 puntos)

De la respuesta en notación O del tiempo de ejecución de los siguientes algoritmos:

I.
```
void silly(int n) {
    for (int i = 0; i < n; ++i) {
       j = n;
       while (j > 0) {
          System.out.println("j = " + j);
          j = j - 2;
       }
    }
}
```

II.
```
void silly(int n, int x, int y) {
    for (int k = n; k > 0; k--)
       if (x < y + n) {
          for (int i = 0; i < n; ++i)
             for (int j = 0; j < i; ++j)
                System.out.println("y = " + y);
       } else {
          System.out.println("x = " + x);
       }
}
```

III.
```
void silly(int n) {
    for (int i = 0; i < n; ++i) {
          for (int j = 0; j < n; ++j)
             System.out.println("j = " + j);
          for (int k = 0; k < i; ++k) {
             System.out.println("k = " + k);
             for (int m = 0; m < 100; ++m)
                System.out.println("m = " + m);
           }
    }
 }
```

IV.
```
int silly(int n, int m) {
    if (m < 2) return m;
    if (n < 1) return n;
    else if (n < 10)
      return silly(n/m, m);
    else
      return silly(n - 1, m);
 }
```

Resolución de Tema Dos:

Tema Tres (15 puntos)

Considere la siguiente implementación de la estructura conjunto (set). Usamos un par de arreglos en la cual un arreglo es más grande y el otro arreglo es más pequeño. Ambos se mantienen ordenados. La longitud del arreglo más pequeño es la raíz cuadrada del arreglo más grande, a pesar de que no esté lleno. Para añadir un elemento, se lo inserta en el orden correspondiente en el arreglo más pequeño, moviendo los elementos existentes si fuera necesario. Si el arreglo más pequeño se llena, se generan un arreglo grande y pequeño de mayor capacidad y se juntan (merge) los dos arreglos en el arreglo más grande. El nuevo arreglo más pequeño ahora está vacío. Para encontrar un elemento usamos búsqueda binaria tanto en el arreglo grande como en el pequeño.

a) Muestre que esta estructura de datos necesita O(lg n) para buscar un elemento (5 p.)
b) Muestre que el tiempo amortizado para insertar un elemento es O($\sqrt{n}$) (10 p.)

Resolución Tema Tres:

Tema Cuatro (10 puntos)

¿Cuál es el tiempo esperado (número de comparaciones) para buscar un elemento que sabemos existe en un arreglo desordenado de N elementos si el algoritmo utilizado es el de búsqueda secuencial (también llamada lineal)? En la búsqueda secuencial se compara el elemento buscado con el primer elemento del arreglo, luego se compara con el segundo, luego con el tercero y así hasta llegar hasta el último. La búsqueda se detiene el momento en que se encuentra al elemento. Considere que los elementos no se repiten.

Resolución Tema Cuatro: