# EXAMEN Estructura de Datos y Algoritmos.
# Julio 2011
## El examen suma 100 puntos. Para aprobar se necesitan 60.

**Ejercicio 1) (15 puntos)**

Calcule el orden del tiempo de ejecución de los siguientes programas:

a)
```
int sum=0;
for (int i=0;i<n;i++)
        for (int j=0;j<n-1;j++)
                sum++;
```

b)
```
int sum=0;
for (int i=1;i<n;i++)
        for (int j=0;j<i;j++)
                sum++;
```

**Ejercicio 2) (15 puntos)**

Implemente el siguiente procedimiento que calcula el máximo y el mínimo de un arreglo de enteros.

```
void max_min(int arreglo [], int & max, int & min)
```

**Ejercicio 3 (30 puntos)**

Considere la siguiente definición del tipo LISTA, de listas de enteros en memoria dinámica:

```
struct nodoLista {
        int info;
        nodoLista *sig;
};
typedef nodoLista *LISTA;
```

*Se pide, sin usar funciones o procedimientos auxiliares, implementar la siguiente función iterativa*:
***IncUno***: Dada una lista *L* de enteros de tipo LISTA, retorna una nueva lista (que no comparte registros de memoria con la lista parámetro) que contiene a todos los elementos de *L* incrementados en 1. Si *L* es una lista vacía, la función debe retornar la lista vacía.

Ejemplos:

| **Entradas** | **Resultado de *IncUno*** |
|---|---|
| L = [2,7,8,3] | [3,8,9,4] |
| L = [2] | [3] |
| L = [] | [] |

**Ejercicio 4 (40 puntos)**

Considere la siguiente definición del tipo ABB de los árboles binarios de búsqueda de enteros, en memoria dinámica:

```
struct nodoABB {
        int dato;
        nodoABB * izq; // menores que dato
        nodoABB * der; // mayores que dato
};
typedef nodoABB *ABB;
```

a) Defina una función recursiva *BorrarMax* que dado un árbol binario de búsqueda de enteros *A* de tipo ABB, no vacío (precondición), elimine y retorne de *A* su máximo elemento. No se pueden usar funciones o procedimientos auxiliares. *BorrarMax* debe evitar recorrer nodos innecesarios de *A*. Deberá liberarse la memoria de la celda cuyo elemento sea eliminado. *A* luego de la función debe ser un árbol binario de búsqueda.

*int BorrarMax (ABB & A)*

b) Defina un procedimiento *BorrarRaiz* que dado un árbol binario de búsqueda de enteros *A* de tipo ABB, no vacío (precondición), elimine el entero que se encuentra en la raíz de *A*. No se pueden usar funciones o procedimientos auxiliares, con excepción de la función definida en la parte (a). *BorrarRaiz* no debe recorrer *A*, aunque si puede hacerlo eventualmente *BorrarMax*. *A* luego del procedimiento debe ser un árbol binario de búsqueda.

*void BorrarRaiz (ABB & A)*