

## Algoritmos y Estructuras de Datos. Parcial 1. [2024-10-15]

- [ATENCIÓN 1]** Para aprobar deben obtener un **puntaje mínimo** del 60 % en las preguntas de teoría y 50 % en clases y operativos.
- [ATENCIÓN 2]** Escribir cada ejercicio en **hoja(s) separada(s)**. Es decir todo **CLAS1** en una o más hojas **separadas**, **OPER1** en una o más hojas **separadas**, **PREG1** en una más hojas **separadas**, etc...
- [ATENCIÓN 3]** Encabezar las hojas con **sección, Nro de hoja (relativo a la sección), apellido, y nombre**,  
**ASI:**

CLAS1, Hoja #2/3	TORVALDS, LINUS
------------------	-----------------

### [Ej. 1] [CLAS1 (W=20pt)]

- [list]**. Escribir la implementación en C++ del TAD **lista** (clase **list**, simplemente enlazada) implementado por punteros ó cursores. Los métodos a implementar son **insert(p,x)**, **erase(p)**, **next()/iterator::operator++(int)** (postfijo), **list()**, **begin()**, **end()**.
- [stack,queue]**. Escribir la implementación en C++ de los métodos **push**, **pop**, **front** y **top** de los TAD pila y cola (clases **stack** y **queue** implementadas con lista), según corresponda.
- [AOO]**. Dada la siguiente implementación de árbol ordenado orientado:

```

1 class cell { ... };                               //< definir
2
3 class iterator_t { ... };                         //< definir
4
5 class tree {
6 private:
7     cell *header;
8     tree(const tree &T);
9 public:
10    tree();
11    ~tree();
12    elem_t &retrive(iterator_t p);
13    iterator_t insert(iterator_t p,elem_t elem); //< implementar
14    iterator_t erase(iterator_t p);
15    void clear();
16    iterator_t begin();
17    iterator_t end();
18    ...
19 }
```

Definir las clases **cell** e **iterator** e implementar el método **insert(...)** de la clase **tree**.

### [Ej. 2] [OPER1 (W=20pt)]

- [Notación  $O(\cdot)$ ]**. cada una de las funciones  $T_1, \dots, T_4$  determinar su velocidad de crecimiento (expresarlo con la notación  $O(\cdot)$ ) y ordenarlas de forma creciente.

$$T_1 = 2n^4 + 4 \cdot 2^n + 2n!$$

$$T_2 = 4.4 \log n + \log(16)n^{3.5} + 2^4$$

$$T_3 = \sqrt[4]{n} + 2 \log_2 n + 2n^3 + 4n^4$$

$$T_4 = 3^2 + 4 \cdot 4^n + 3n^3$$

$$T_5 = \log_2 n + 4$$

- [operaciones]**.

Dada la lista de enteros **L**, ordenada en forma creciente del 1 al 10, la pila **S** y la cola **Q**, ambas vacías. Se ejecuta el siguiente código,

```

1 auto it = L.begin();
2 while (it != L.end()) {
3     if (*it % 2 == 0) {
4         Q.push(*it);
5         it++;
6     } else {
7         S.push(*it);
8         it = L.erase(it);
9     }
10 }

```

(2.b.1) Indicar el contenido resultante de L, S y Q

(2.b.2) Indicar los valores que devuelven S.top() y Q.front().

(2.c) [maps]. Dada la siguiente correspondencia

```
1 map<string,list<int>> M = {{ "Ana",{85, 90}}, {"Luis",{78, 82}}, {"Eva",{91, 89}}};
```

y las siguientes operaciones,

(2.c.1) indicar el efecto de cada operación,

```

1 M.clear();
2 M["Mario"];
3 M["Ana"] = {93};
4 M["Eva"].push_back(95);
5 M["Eva"].clear();
6 M.erase("Ana");

```

(2.c.2) indicar el estado final de la correspondencia.

(2.d) [list-iter].

```

1 list<int> L = {4,6,3,5,4,1,0,7,6,2};
2 auto p = find(L.begin(),L.end(),5);
3 auto q = p++;
4 p = q--;

```

Que valores tienen \*p y \*q?

### [Ej. 3] [PREG1 (W=20pt)]

(3.a) Considerando el árbol (X (R O P) (A (Z (B C (D W)))) decir cuál son los nodos **descendientes(R)**, **antecesores(R)**, **izquierda(R)** y **derecha(R)**.

(Nota: se refiere a antecesores y descendientes **propios**).

(3.b) Sea un árbol (AOO) T=(5 (9 0 2) 1 6) ¿Cómo queda T si hacemos la siguiente inserción:

```

1 auto p = T.find(1);
2 T.insert(p,4);

```

(3.c) En una **correspondencia**:

(3.c.1) ¿Puede haber una **misma clave** con **diferentes valores**? Por ej. M=(3->7, 3->8)

(3.c.2) ¿Puede haber **diferentes claves** con el **mismo valor**? Por ej. M=(2->3, 5->3)

(3.d) Defina que es un **camino** en un árbol. Dado el AOO (A (R S) (Z X W) U), cuáles de los siguientes son caminos?

(3.d.1) [S R A],

(3.d.2) [A Z W],

(3.d.3) [R A Z X],

(3.d.4) [A R S].

(3.e) ¿Cuáles son los tiempos de ejecución para los siguientes métodos de la clase map<> implementada con listas desordenadas en el caso promedio? Métodos: **find(key)**, **M[key]**, **erase(key)**, **erase(p)**, **begin()**, **end()**, **clear()**.

(3.f) Explique que quiere decir la propiedad de **transitividad** de la notación  $O()$ .