Apellido y Nombre: _____

Carrera: _____ DNI: _____

[Llenar con letra mayúscula de imprenta GRANDE]

Universidad Nacional del Litoral
Facultad de Ingeniería y Ciencias Hídricas
Departamento de Informática
Algoritmos y Estructuras de Datos

Algoritmos y Estructuras de Datos.
Recup Parciales. [2023-11-23]

1. **[ATENCIÓN 1]** Para aprobar deben obtener un **puntaje mínimo** del 60 % en las preguntas de teoría y 50 % en clases y operativos.
2. **[ATENCIÓN 2]** Escribir cada ejercicio en **hoja(s) separada(s)**. Es decir todo `CLAS2` en una o más hojas **separadas**, `OPER2` en una o más hojas **separadas**, `PREG2` en una más hojas **separadas**, etc...
3. **[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)]

a) **[list]** Escribir la implementación en C++ del TAD lista (clase **`list`**) implementado por punteros ó cursores. Los métodos a implementar son
   1) **`insert(p,x)`**,
   2) **`erase(p)`**,
   3) **`next()/iterator::operator++(int)(postfijo)`**,
   4) **`list()`**,
   5) **`begin()`**,
   6) **`end()`**.

b) **[stack/queue]** Escribir la implementación en C++ de los métodos
   1) **`push`**,
   2) **`pop`**,
   3) **`front`**,
   4) **`top`**,

   de los TAD pila y cola (clases **`stack`** y **`queue`**), según corresponda.

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

a) **[AOO]**: Dado el árbol ordenado orientado codificado en Lisp como **`(T Z (R (W X Y) F) Q)`**.
   1) Indicar su altura. Justifique
   2) Indique el nivel del nodo **`R`**.
   3) Determinar la partición respecto del nodo **`R`**.
   4) Indique su recorrido en preorden.
   5) Indique su recorrido en postorden

b) **[operaciones]**: Sea el árbol ordenado orientado **`D=(5 (4 3 2) (1 9 8 7) (6 5))`**.
   1) Grafique el árbol inicial.
   2) Indique como queda el ábol **`D`** después de hacer:

```
auto n = D.find(2);
n = D.insert(n,0);
```

   3) Indique como queda el ábol **`D`** después de hacer:

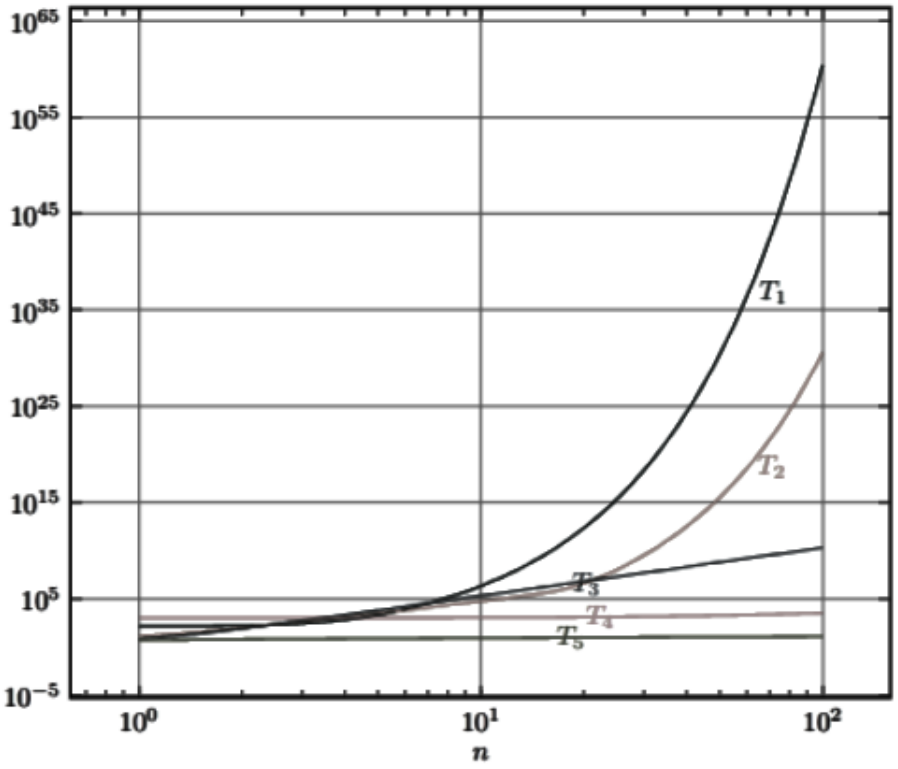
```
n++;
n = n.lchild();
n = D.insert(n,10);
```

4) Indique `D` después de hacer (siempre sobre el árbol resultado de la operación anterior, es decir las operaciones son acumulativas):

```
1 n = D.find(1);
2 n = D.remove(n)
3 D.insert(n,99);
```

c) **[bigO]**: Asocie la velocidad de crecimiento correspondiente a las tareas cuyo crecimiento se indica en la siguiente gráfica.

1) $O(n^5)$
2) $O(4^n)$
3) $O(\log(n))$
4) $O(2^n)$
5) $O(\log(2)n^{3/2})$



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

a) ¿Cuantas posiciones **no dereferenciables** puede haber en una **lista**? ¿Y en un **árbol**?
b) Supongamos que tenemos un árbol (AOO) `T=(6 (0 1 3) 2 7)` y un iterator `p` apuntando al 2. ¿Cómo queda `T` si hacemos una inserción: `T.insert(p,9);`
c) En una **correspondencia**: ¿puede haber una **misma clave** con **diferentes valores**? Por ej. `M=(3->5, 3->6)` ¿Puede haber **diferentes claves** con el **mismo valor**? Por ej. `M=(2->1, 6->1)`
d) Sea el árbol `(p (z o x) (y (n r s)))`. Cuáles de los siguientes son **caminos**?

- `(n y p)`
- `(p y n)`

- `(y p z x)`
- `(y n s)`

e) ¿Cuál es el **tiempo de ejecución** (mejor/promedio/peor) de las siguientes funciones? (asumir listas implementadas con **celdas enlazadas por punteros o cursores**)

1) `list<T>::begin()`,
2) `list<T>::insert(p,x)`,
3) `list<T>::clear(p,x)`,
4) `map<K,V>::find(x)`, para la implementación con vectores ordenados,
5) `map<K,V>::find(x)`, para la implementación con listas ordenadas.

f) ¿Como se define la **notación asintótica** $T(n) = O(f(n))$? ¿Porqué decimos que $(n+1)^2 = O(n^2)$ si siempre es $(n+1)^2 > n^2$?

[Ej. 4] [CLAS2 (W=20pt)]

a) **[ab]** declarar las clases `btree`, `cell`, `iterator`, (preferentemente respetando el anidamiento), incluyendo las declaraciones de datos miembros. Implementar el método `btree<T>::iterator btree<T>::erase(btree<T>::iterator n)`

b) **[set]** Implemente los métodos `set_union`, `set_intersection` y `set_difference` que realizan las operaciones binarias correspondientes entre los sets `A` y `B`, almacenando el resultado en `C`.

[Ej. 5] [OPER2 (W=20pt)]

a) **[huffman]**: Dado el árbol indicado por el siguiente código Lisp: `(. L (. T (. N (. (. G E) A)`

1) Indique el código correspondiente a cada símbolo (izq = 0, der = 1).
2) Decodifique: `01110011111101011101`
3) Considerando probabilidades `(A = 0.10, E = 0.04, G = 0.01, L = 0.40, N = 0.15, T = 0.30,)`, calcule la longitud media del código.

b) **[hash]**: En una tabla de dispersión cerrada con `B=5` cubetas, con función de dispersión y redispersión lineal. (Mostrar la tabla resultante de cada inciso).

1) Insertar los enteros `{2, 7, 5, 12, 7, 9, 2}`
2) Luego eliminar los elementos 12 y 5.
3) Insertar el elemento 38.

c) **[ABB]**: Construir un árbol binario de búsqueda (indicar el árbol tras cada modificación):

1) insertando sucesivamente los elementos: `{8, 10, 3, 6, 4, 1,9,15}`.
2) Eliminar 8.
3) Insertar 13.
4) Insertar 9.

[Ej. 6] [PREG2 (W=20pt)]

a) Explique cual es la condición de códigos prefijos. De un ejemplo de códigos que cumplen con la condicion de prefijo y que no cumplen para un conjuntos de 3 caracteres.

b) Exprese como se calcula la longitud promedio de un código de Huffman en función de las probabilidades de cada unos de los caracteres P_i, de la longitud de cada caracter L_i para un número N_c de caracteres a codificar.

c) Se quiere representar el conjunto de enteros múltiplos de 5 entre 10 y 95 (o sea $U = \{10, 15, 20, ..., 95\}$) por **vectores de bits**, escribir las funciones **`indx()`** y **`element()`** correspondientes.

d) Discuta la complejidad algorítmica de las operaciones binarias **`set_union(A,B,C)`**, **`set_intersection(A,B,C)`**, y **`set_difference(A,B,C)`** para conjuntos implementados por vectores de bits, donde **`A`**, **`B`**, y **`C`** son subconjuntos de tamaño n_A, n_B, y n_C respectivamente, de un conjunto universal **`U`** de tamaño N.

e) ¿Cuál es el costo de **inserción exitosa** en tablas de dispersión abiertas?

f) Si la correspondencia **`map<int,string> M`** contiene **`M={5->"Paredes",10->"Messi"}`** y ejecutamos el código **`string z= M[5]`**. ¿Que valor toma **`z`**? ¿Cómo queda **`M`**? Si ahora hacemos **`z = M[8]`**, ¿cómo quedan **`z`** y **`M`**?

g) Si queremos generar un código binario de longitud fija para el conjunto de letras minúsculas y dígitos (en total 26+10=36 caracteres. ¿Cuantos bits tendrá, como **mínimo**, la representación de cada caracter? ¿Y para el conjunto de caracteres que son dígitos (en total 10)?

h) ¿Es posible **insertar** en una posición **no-dereferenciable** (Λ) en un árbol binario (AB)? ¿Y en una posición **dereferenciable**?