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.
Parcial 2. [2019-11-14]

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**: CLAS2, Hoja #2/3 TORVALDS, LINUS

[Ej. 1] [CLAS2 (W=20pt)]

a) Defina la estructura de la clase celda de un AB (`btree<T>::cell`). Presente los atributos y el prototipo de los métodos principales.

b) Defina la estructura de la clase iterador de un AB (`btree<T>::iterator`). Presente los atributos y el prototipo de los métodos principales.

c) Dado el siguiente fragmento de código:

```
1 btree<char> T;
2 lisp2btree("(a (h k .) m)", T);
3 btree<char>::iterator p = T.begin().left().right();
```

En base a *a)* y *b)*, presente un croquis de todas las estructuras `btree<T>::cell` e `btree<T>::iterator` presentes en memoria luego de la ejecución y los punteros que las relacionan.

d) Implementar una función
`bool openhashtable_insert(vector<list<T> >& table, unsigned int (*hashfunc)(T), T x)`
que inserta el elemento `x` en la tabla de dispersión abierta `table` utilizando la función de dispersión `hashfunc` y retorna un booleano indicando si la inserción fue o no exitosa.

[Ej. 2] [OPER2 (W=20pt)]

a) **[huffman]**: Dados los caracteres siguientes con sus correspondientes probabilidades, construir el código binario utilizando el algoritmo de Huffman y encodar la palabra `CYBERMONDAY`, $P(O) = 0.10, P(D) = 0.10, P(B) = 0.10, P(N) = 0.30, P(C) = 0.10, P(A) = 0.02, P(E) = 0.02, P(R) = 0.02, P(Y) = 0.04, P(M) = 0.20$. Calcular la longitud promedio del código obtenido.

b) **[hf-decode]** Utilizando el código Lisp
`(. (. (. Y A) (. (. C L) ␣)) (. D (. R (. (. F K) (. I B)))))`
desencodar el mensaje:
`11111010100101001110101111100110111101000 1000`.

c) **[abb]** Dados los enteros $\{14, 8, 21, 3, 4, 11, 6, 5, 4, 13, 2\}$ insertarlos, en ese orden, en un **árbol binario de búsqueda (ABB)**. Mostrar las operaciones necesarias para eliminar los elementos 14, 8 y 3 en ese orden.

d) **[hash-dict]** Insertar los números $\{200, 180, 280, 110, 100, 380, 200, 700\}$ en una **tabla de dispersión cerrada** con $B = 10$ cubetas, con función de dispersión **h(x) = x/(B*10)** y redispersión lineal. Muestre la tabla resultante. Luego elimine el elemento 110 y muestre la tabla final.

e) **[quick-sort]** Dados los enteros $\{6, 10, 5, 2, 6, 11, 9, 4, 4, 3, 12, 7\}$ ordenarlos por el método de *clasificación rápida (quick-sort)*. En cada iteración indicar el pivote y mostrar el resultado de la partición. Utilizar la estrategia de elección del pivote discutida en el curso, a saber el mayor de los dos primeros elementos distintos.

[Ej. 3] [PREG2 (W=20pt)]

a) Comente ventajas y desventajas de las **tablas de dispersión abiertas y cerradas**. Además analice costos de inserción y remoción en cada una de ellas.

b) Se quiere representar conjuntos de **cuadrados perfectos** en el rango **`[0,100]`** es decir por ejemplo **`0,4,9,16,25,...,100`**, por **vectores de bits**. ¿Cuál es el tamaño **`N`** del conjunto universal? Completar las funciones **`indx()`** y **`element()`** correspondientes:

```
1       int indx(int p) { /* ... */ }
2       int element(int z) { /* ... */ }
```

c) Si queremos generar un código binario de **longitud fija** para el conjunto de pares de letras (por ejemplo **`df`**, **`qz`**...), o sea en total **`26*26=676`** "caracteres". ¿Cuantos bits tendrá, como **mínimo**, la representación de cada caracter? ¿Y para el conjunto de pares de caracteres mayúsculas y dígitos (**`52*52=2704`** caracteres)?

d) Discuta la **estabilidad** del algoritmo de ordenamiento de **listas por fusión** (**merge-sort**). ¿Es estable? ¿Bajo que condiciones?

e) ¿Cuál es el tiempo de ejecución del algoritmo de **ordenamiento rápido** (**quick-sort**), en el caso promedio y en el peor caso? ¿Cuando se produce el peor caso?

f) ¿Qué dos condiciones cumple un montículo? Explique la utilidad de cada condición.