

Algoritmos y Estructuras de Datos. 3er Parcial. Tema: 1A. [22 de junio de 2006]

[Ej. 1] [clases (20 puntos)]

- a) [merge-sort (5pts)] Implemente el algoritmo de ordenamiento por fusión para listas `void merge_sort(list<int> &L, bool (*comp)(int,int))`. Asuma que dispone de las funciones auxiliares (NO es necesario implementarlas):
- `split(L,L1,L2)` que separe una lista L en dos listas L1 y L2, dejando a L vacía.
 - `join(L1,L2,L,comp)` que fusiona las listas **ordenadas** L1 y L2 en una lista **ordenada** L utilizando la **función de comparación comp**, dejando a L1 y L2 vacías.
- b) [set-abb (15pts)] Implemente los siguientes métodos de la clase `set<>` por árbol binario de búsqueda: `begin()`, `find(x)`, `insert(x)`.

[Ej. 2] [programación (total = 45 puntos)]

- a) [es-simétrica (15 puntos)]. Considere la correspondencia de adyacencia simétrica `map<int,set<int>>> G`, la cual asigna a cada vértice de un grafo, el subconjunto de sus vértices adyacentes (o primera capa de vecinos). Por ejemplo, la correspondencia de adyacencias G_1 para el grafo G de 6 vértices y 7 aristas mostrado en la Fig. es la indicada.

0 --> {1, 4, 5}	0---1 2
1 --> {0, 4}	G \ /
2 --> {3, 4}	\ /
3 --> {2, 4}	\ /
4 --> {0, 1, 2, 3}	5 4---3
5 --> {0}	

Al decir que G es una correspondencia simétrica, significa decir que para cada arista (i,j) presente en G (con $i \neq j$), entonces también debería estar presente la arista (j,i) , para todo $i,j \in V$, donde V es el conjunto de vértices del grafo. Pero, podría suceder, que la correspondencia ingresada no sea completamente simétrica, por ejemplo, la correspondencia de adyacencias G_2 no es completamente simétrica.

```

0 --> {1, 4}
1 --> {0, 4}
2 --> {4}
3 --> {2, 4}
4 --> {0, 2, 3}
5 --> {0}

```

Consigna: escribir la función `bool es_simetrica (map<int,set<int> > &G)` que devuelva `true` si G es una correspondencia de adyacencia simétrica y falso en caso contrario.

- b) [de-arista-a-adyacencia (15 puntos)]. Suponga que se ingresan las aristas de un grafo G mediante un `list<pair<int,int>> L` de pares de enteros, donde cada par representa los vértices de cada arista. Por ejemplo, el grafo F lo describimos mediante la lista de aristas $L = \{(0,2), (2,5), (3,4), (4,1), (0,3), (2,4), (5,1), (0,4), (5,4), (6,5), (6,1)\}$, por lo que la correspondencia de adyacencias simétrica `map<int,set<int>>> F` estará dada por

0 --> {2, 3, 4}	0---2---5---6
1 --> {4, 5, 6}	\ / /
2 --> {0, 4, 5}	\ / / F
3 --> {0, 4}	\ / /
4 --> {0, 1, 2, 3, 5}	\ / /
5 --> {1, 2, 4, 6}	3---4---1
6 --> {1, 5}	

Apellido y Nombre: _____

Carrera: _____ DNI: _____

[Llenar con letra mayúscula de imprenta GRANDE]

Consigna: escribir la función

`void de_arista_a_adyacencia (list<pair<int,int>> &L, map<int,set<int>> &F)` la cual, dada la lista `L` de aristas, devuelva la correspondencia de adyacencia *simétrica* `F`.

- c) **[cuenta-aristas (15 puntos)]**. Considere la correspondencia de adyacencia simétrica `G <int,set<int,int>>`, la cual asigna a cada vértice de un grafo, el subconjunto de sus vértices adyacentes (o primera capa de vecinos).

Consigna: escribir la función `int cuenta_aristas(map<int,set<int> > &G)` que devuelve el número de aristas presentes en la correspondencia de adyacencia simétrica `G`. Por ejemplo, para el grafo `G` debe retornar 7 y para el grafo `F` debe retornar 11.

[Ej. 3] [operativos (total = 25 puntos)]

- a) **[heap-sort (10 pts)]** Dados los enteros $\{0, 4, 7, 1, 2, 12, 9\}$ ordenarlos por el método de “montículos” (“*heap-sort*”). Mostrar el montículo (minimal) antes y después de **cada** inserción/supresión.
- b) **[open-hash (5 pts)]** Considere una tabla de dispersión **abierta** con 4 cubetas para almacenar strings con la función de dispersión `int hash(std::string &s) { return s.size(); }`. Inserte las palabras de la frase TRE GRANDES TIGRES TRAGABAN TRES GRANDES PLATOS DE TRIGO.
- c) **[abb (5 pts)]** Dados los enteros $\{12, 6, 19, 1, 2, 9, 4, 3, 0, 11\}$ insertarlos, en ese orden, en un “árbol binario de búsqueda”. Mostrar las operaciones necesarias para eliminar los elementos 12, 11. en ese orden.
- d) **[hash-dict (5 pts)]** Insertar los números 0, 13, 23, 6, 5, 33, 15, 2, 25 en una tabla de dispersión cerrada con $B = 10$ cubetas, con función de dispersión $h(x) = x$ y estrategia de redispersión lineal.

[Ej. 4] [Preguntas (total = 10 puntos)] Responder según el sistema “multiple choice”, es decir marcar con una cruz el casillero apropiado. **Atención:** Algunas respuestas son intencionalmente “descabelladas” y tienen puntajes **negativos!!**

- a) ¿Cuál de los siguientes es el resultado correcto de ordenar la secuencia $\{-3, 5, 3, 2, -2, 1, 2, -3\}$ por la relación de orden débil $|a| < |b|$?

- ☐ ... $\{1, 2, -2, 2, -3, 3, -3, 5\}$
☐ ... $\{-3, -3, -3, 1, 2, 2, 3, 5\}$
☐ ... $\{1, 2, 2, -2, 3, -3, -3, 5\}$
☐ ... $\{1, -2, 2, 2, -3, -3, 3, 5\}$

- b) ¿Cuál es el número de *intercambios* en el método de clasificación por selección?

- ☐ ... $O(\log n)$
☐ ... $O(n)$
☐ ... $O(1)$
☐ ... $O(n^2)$

- c) El tiempo de ejecución de “quick-sort” en el peor caso es

- ☐ ... $O(n)$
☐ ... $O(1)$
☐ ... $O(n^2)$
☐ ... $O(\log n)$