Ejercitación – Heap

Este ejercicio consiste en implementar una cola de prioridad ColaPrior<T> con elementos de tipo T, utilizando como estructura un heap implementado sobre un arreglo (std::vector).

El projecto posee un único target: correrTests. Puede ejecutarse desde CLion o utilizando cmake y makefile como de costumbre.

Completar la implementación de los siguientes métodos de la clase ColaPrior<T> en el archivo src/ColaPrior.hpp:

- 1. ColaPrior() Construye una cola vacía.
- 2. int tam() const Devuelve la cantidad de elementos en la cola.
- 3. void encolar(const T& elem) Encola un elemento. La cola puede tener elementos repetidos más de una vez.

Observación: dado que la cola se implementa usando un std::vector esta operación es O(n) en peor caso, ya que potencialmente se debe redimensionar la estructura de std::vector. Sin embargo, esta operación debería tener costo $O(\log n)$ si la operación $push_back$ de std::vector fuera O(1). Más precisamente, el costo amortizado debe ser $O(\log n)$, es decir, el costo de hacer n inserciones debe ser $O(n\log n)$ en peor caso.

- 4. const T& proximo() const Devuelve el elemento de mayor prioridad. *Precondición*: debe haber al menos un elemento en la cola.
- 5. void desencolar() Saca el elemento de mayor prioridad. Igual que en el caso de encolar, esta operación debe tener costo $O(\log n)$ amortizado. Precondición: debe haber al menos un elemento en la cola.
- 6. ColaPrior(const vector<T>& elems) Construye una cola que contiene todos los elementos del vector utilizando el algoritmo heapify.