7) La implementación basada en vector podría ser eficiente si tomamos como principio la cola del vector, es decir, si utilizamos las operaciones pop_back y push_back. Del otro modo si insertamos al principio del vector, considerando un vector con n elementos habría que mover estos n elementos un espacio a la derecha. Y eliminarK en peor caso cuando se hace eliminarK(1) habría que desplazar n-1 elementos a la izquierda. Para ambas operaciones el análisis asintotico es O(n).

La implementación basada en List es eficiente para todas las funciones que se piden. Se puede usar tanto el final como el principio de la list ya que ambas operaciones son realizadas en O(1), lo mismo pasa con eliminar ya que la list es basada en Doubly-linked list. El eliminar n elementos también se podría conseguir en tiempo constante ya que se utilizan punteros.

Para la implementación basada en stack la inserción se realiza en tiempo constante, al igual que la eliminación a un elementos. Para eliminarK(n) sería O(n) el peor caso. ~~(Al igual que en implementación vector)~~.

La implementación basada en Queue ND es eficiente. Si bien la inserción es O(1) para la eliminación, que en el peor caso para la implementación sería eliminarK(1) habría que mover n-1 elementos a una Queue auxiliar, hacer la eliminación del elemento n y luego devolver estos n-1 elementos a la Queue original por ende eliminarK() para esta implementación es O(n).

Para la implementación basada en Deque pasa similar a la list. Se puede considerar ambos extremos ya que tanto insertar como eliminar es en tiempo constante eso sí, la eliminación se hace uno a uno por lo que eliminarK(n) es O(n).

La secuencia al ser algo más general que las anteriores sus operaciones de insertar y remover un elemento al principio o al final es de tiempo constante. Sin embargo como ocurre con otras implementaciones eliminarK(n) es O(n) ya que se realiza de uno en uno