10 – TAD Lista 2 de 2

DOCENTE – FEDERICO VILENSKY

TADs acotados y no acotados

- Los TADs pueden ser acotados o no en la cantidad de elementos
  - Acotado significa que se puede llenar
  - No acotado es que puede ser arbitrariamente grande
- En general para acotados vamos a utilizar implementaciones estáticas
- Y para los no acotados vamos a utilizar implementaciones dinámicas
- Es importante tener esta noción, porque son conceptualmente distintos

TADs acotados y no acotados

- Los TADs acotados van a tener una operación que nos indica si la estructura está llena
- `virtual bool EstaLlena()`
- Esto además nos va a generar una precondición a la hora de agregar elementos al TAD
  - Vamos a tener que chequear que la estructura no esté llena
- En C++ lo vamos a estar viendo como una subclase de la no acotada, esto no quita que sean distintas

TADs acotados y no acotados

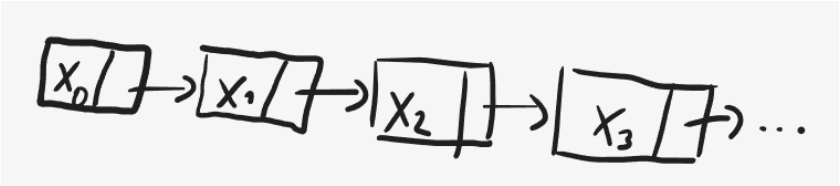
- Si bien son cosas distintas los TADs acotados y no acotados
- Los vamos a unificar en una única implementación, de la siguiente manera
  - Incluimos las operaciones del TAD no acotado
  - Le agregamos la operación `virtual bool EstaLlena()`, que nos dice si la estructura se llenó
    - Las no acotadas siempre van a devolver false
  - Las operaciones de inserción pasan a tener como precondición que la estructura no esté llena
    - Como las no acotadas siempre nos dicen que NO están llenas, siempre van a cumplir esta precondición

TADs acotados y no acotados

- Con esto logramos definir un único TAD para tanto la versión acotada, como la no acotada
- Esto es algo que podemos hacer porque el comportamiento (sin contar que se puede llenar) es el mismo en ambos casos

Implementaciones de Listas

- En teórico vimos una sola representación de las Listas, estas siendo listas encadenadas simples

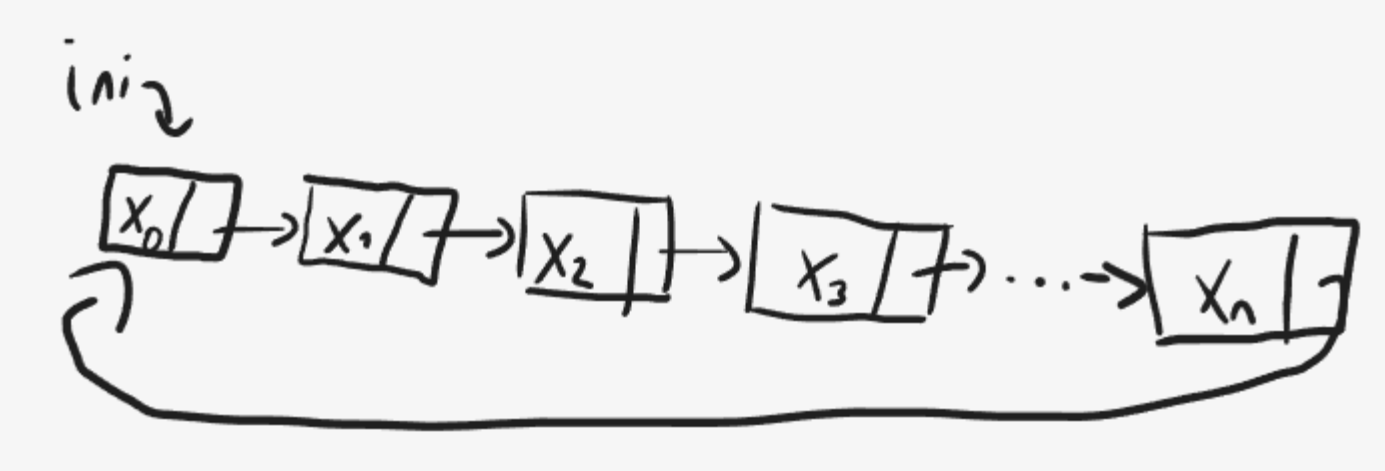


- También van a ver otras implementaciones
  - Dinámicas
    - Doblemente encadenada
    - Circular
  - Estáticas
    - Con arrays (ArrayList)

Ilustraciones

- Doblemente encadenada:

- Circular:



Ilustraciones

- Con arreglos

Ejercicio – Lista indexada

- Considere el TAD Lista indexada genérico de elementos de tipo T
- Con las operaciones
  - ListaVacia: construye una lista vacia
  - Insertar: dado un entero n y un elemento elem de tipo T. Inserta a elem en la lista en la posición n
    - Si el tamaño de la lista es mas chico que n-1, inserta elem al final de la lista
    - Se el tamaño de la lista es mayor o igual a n-1, inserta elem en la posición n, y mueve el resto hacia la derecha
  - EstaVacia: devuelve true sii la lista está vacía
  - Pertenece: recibe un elemento elem de tipo T, devuelve true sii elem pertenece a la lista
  - ElementoEn: recibe un entero n, y devuelve el elemento en la posición n
    - Pensar que pasa si n es mas grande que la lista
  - Borrar: recibe un entero n, y borra el elemento en esa posición
    - Si la lista es más chica que n no hace nada
    - Si la lista tiene al menos n elementos, elimina al elemento en la posición n, y mueve al resto de los elementos que le siguen hacia la izquierda

Ejercicio – Listas indexadas

1. Especificar el TAD ListasIndexadas, no acotado, de tipo genérico T
2. Implementar el TAD usando memoria dinámica (nodos y punteros)
3. Generalizar ListasIndexadas, para que sea no necesariamente acotado
4. Pensar, e implementar una implementación acotada
5. Estudiar el orden de tiempo de ejecución de las operaciones en ambas implementaciones