

Abstracción de Datos y Listas Encadenadas

Programación de Estructuras de Datos y Algoritmos Fundamentales
(TC1031)

M.C. Xavier Sánchez Díaz
sax@tec.mx



Outline

- 1 Abstracción de Datos
 - Diseñando Estructuras de Datos
- 2 Listas vinculadas (*Linked lists*)
- 3 Pilas (*Stack*)
- 4 Colas (*Queue*)

Tipos de datos

Abstracción de Datos

A lo largo del curso hemos usado **tipos de datos** (*datatypes*) para nuestros vectores y arreglos.

Estos **tipos de datos** han sido para representar los siguientes datos:

- `int` o números enteros
- `float` o números flotantes (decimales)
- `string` o cadenas de caracteres ('palabras')

Y cada uno de ellos tiene un *operaciones válidas* y un *rango* asociados.

Tipos de datos

Abstracción de Datos

A lo largo del curso hemos usado **tipos de datos** (*datatypes*) para nuestros vectores y arreglos.

Estos **tipos de datos** han sido para representar los siguientes datos:

- `int` o números enteros
- `float` o números flotantes (decimales)
- `string` o cadenas de caracteres ('palabras')

Y cada uno de ellos tiene un *operaciones válidas* y un *rango* asociados.

Tipos de datos

Abstracción de Datos

A lo largo del curso hemos usado **tipos de datos** (*datatypes*) para nuestros vectores y arreglos.

Estos **tipos de datos** han sido para representar los siguientes datos:

- `int` o números enteros
- `float` o números flotantes (decimales)
- `string` o cadenas de caracteres ('palabras')

Y cada uno de ellos tiene un *operaciones válidas* y un *rango* asociados.

Tipos de datos

Abstracción de Datos

A lo largo del curso hemos usado **tipos de datos** (*datatypes*) para nuestros vectores y arreglos.

Estos **tipos de datos** han sido para representar los siguientes datos:

- `int` o números enteros
- `float` o números flotantes (decimales)
- `string` o cadenas de caracteres ('palabras')

Y cada uno de ellos tiene un *operaciones válidas* y un *rango* asociados.

Datos más complicados

Abstracción de datos

¿Qué pasa si en lugar de números, quiero trabajar con *otra cosa*?

Por ejemplo, a una coordenada en formato (x, y) puedo *sumarle* una magnitud y debería obtener *otra* coordenada.

En este caso, las **coordenadas** son un *dato estructurado*:

- Tiene una x
- Tiene una y
- Tiene una representación textual en formato (x, y)
- Puedo sumarle (y restarle) magnitudes para obtener otras **coordenadas**

Ésta es una **estructura de datos**.

Datos más complicados

Abstracción de datos

¿Qué pasa si en lugar de números, quiero trabajar con *otra cosa*?

Por ejemplo, a una coordenada en formato (x, y) puedo *sumarle* una magnitud y debería obtener *otra* coordenada.

En este caso, las *coordenadas* son un *dato estructurado*:

- Tiene una x
- Tiene una y
- Tiene una representación textual en formato (x, y)
- Puedo sumarle (y restarle) magnitudes para obtener otras *coordenadas*

Ésta es una *estructura de datos*.

Datos más complicados

Abstracción de datos

¿Qué pasa si en lugar de números, quiero trabajar con *otra cosa*?

Por ejemplo, a una **coordenada** en formato (x, y) puedo *sumarle* una magnitud y debería obtener *otra coordenada*.

En este caso, las **coordenadas** son un *dato estructurado*:

- Tiene una x
- Tiene una y
- Tiene una representación textual en formato (x, y)
- Puedo sumarle (y restarle) magnitudes para obtener otras **coordenadas**

Ésta es una **estructura de datos**.

Datos más complicados

Abstracción de datos

¿Qué pasa si en lugar de números, quiero trabajar con *otra cosa*?

Por ejemplo, a una **coordenada** en formato (x, y) puedo *sumarle* una magnitud y debería obtener *otra coordenada*.

En este caso, las **coordenadas** son un *dato estructurado*:

- Tiene una x
- Tiene una y
- Tiene una representación textual en formato (x, y)
- Puedo sumarle (y restarle) magnitudes para obtener otras **coordenadas**

Ésta es una **estructura de datos**.

Datos más complicados

Abstracción de datos

¿Qué pasa si en lugar de números, quiero trabajar con *otra cosa*?

Por ejemplo, a una *coordenada* en formato (x, y) puedo *sumarle* una magnitud y debería obtener *otra coordenada*.

En este caso, las *coordenadas* son un *dato estructurado*:

- Tiene una x
- Tiene una y
- Tiene una representación textual en formato (x, y)
- Puedo sumarle (y restarle) magnitudes para obtener otras *coordenadas*

Ésta es una *estructura de datos*.

Story time: **structs** vs **objects**

*Story time: Old Style vs New
Style*

Story time: structs vs objects

*Story time: Old Style vs New
Style*

Niveles de abstracción de datos

Abstracción. Cuando organizamos los datos de alguna manera **lógica** o *abstracta*—general—que después pueda ser aterrizada a distintas implementaciones. El ‘**qué**’ se explica en este nivel.

Implementación. En este nivel se definen e **implementan** las operaciones posibles que se le pueden aplicar a la estructura. El ‘**cómo**’ se explica en este nivel.

Aplicación. Se aterriza dicha estructura a un uso específico por medio de una **instancia** y se utiliza para resolver el problema correspondiente. El ‘**dónde**’ o ‘**con qué**’ se explica en este nivel.

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una matriz tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una matriz tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una matriz tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una matriz tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del **cómo**

Ejemplo: una **matriz** tiene

- **Celdas** como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una **matriz** tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una **matriz** tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

¿Qué tiene una estructura de datos *Old Style*?

Abstracción de Datos

- Los **elementos** de la estructura son parte del 'qué'
- La **organización** de los elementos de la estructura
- El **dominio** de los datos de la estructura
- Las **operaciones** que pueden aplicarse a la estructura son parte del cómo

Ejemplo: una **matriz** tiene

- Celdas como **elementos**
- Se **organiza** en renglones y columnas
- Puede tener los **valores** que nosotros determinemos
- Se le pueden **sumar/restar/multiplicar** escalares o con otras matrices (bajo *ciertas condiciones*)

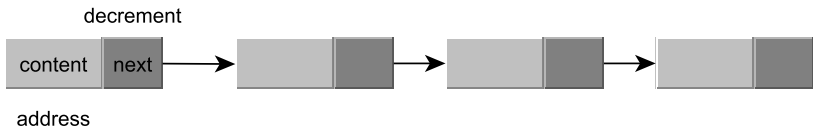
Disclaimer 1: *Todo es matemáticas.
Esto no se limita a Ciencias
Computacionales*

Disclaimer 2: *Todo se hace ahora
con objetos¹*

¹Estamos en el siglo XXI, come on. . .

Listas vinculadas

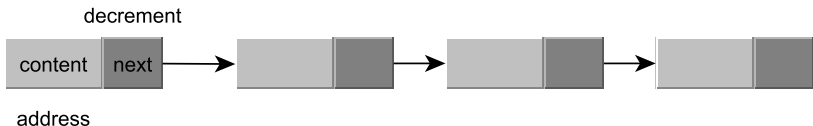
Una **lista vinculada** (*linked list*)—o simplemente *lista*—es una estructura de datos **recursiva** que tiene una organización lineal (como la de un arreglo) y donde cada **nodo** tiene dos ‘celdas’: una dirección (contenido) y un decremento (un apuntador a otra celda).



Story time: CAR vs. CDR

Listas vinculadas

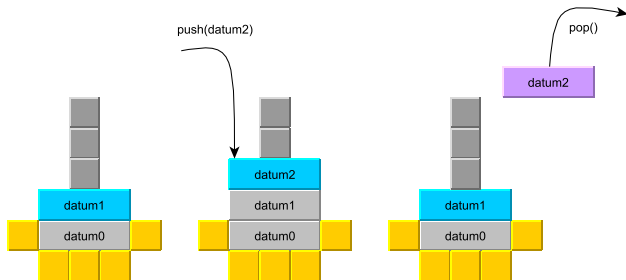
Una **lista vinculada** (*linked list*)—o simplemente *lista*—es una estructura de datos **recursiva** que tiene una organización lineal (como la de un arreglo) y donde cada **nodo** tiene dos ‘celdas’: una dirección (contenido) y un decremento (un apuntador a otra celda).



Story time: CAR vs. CDR

Pilas

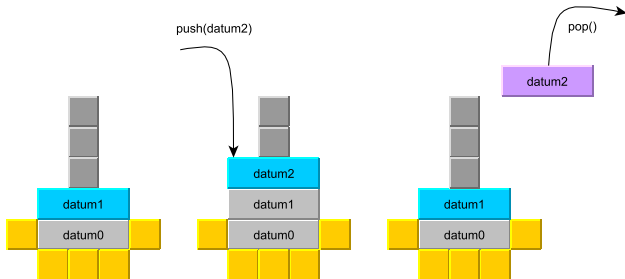
Una **pila** (*stack*) es una estructura de datos de organización lineal (como la de un arreglo o una lista vinculada), en donde vamos **apilando** los elementos. Como un montón de platos apilados, el que sacas (a menos que estés demente) suele ser el de hasta arriba (que es el último que entró): FILO mode.



¿A qué crees que se refiere el famoso *Stack Overflow*?

Pilas

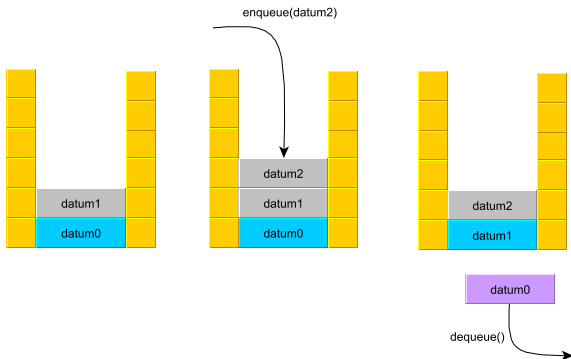
Una **pila** (*stack*) es una estructura de datos de organización lineal (como la de un arreglo o una lista vinculada), en donde vamos **apilando** los elementos. Como un montón de platos apilados, el que sacas (a menos que estés demente) suele ser el de hasta arriba (que es el último que entró): FILO mode.



¿A qué crees que se refiere el famoso *Stack Overflow*?

Colas

Una **cola** (*queue*) es una estructura de datos de organización lineal (como la de un arreglo o una lista vinculada), en donde vamos **enfilando** los elementos. Como un montón de personas enfiladas, el que pasa primero (a menos que tenga paros) suele ser el que está hasta adelante (que es el primero que se formó): FIFO mode.



Quiz time!



Creeper stack or Creeper queue?



SoShi stack or SoShi queue?

```
sizeof(mylist.push(mynumber + 5))
```

Function stack or function queue?


```
mynumber = mynumber + 5;  
mylist.push(mynumber);  
mylist.getSize();
```

Function stack or function queue?