

Lists

CSE1303 Part A

Data Structures and Algorithms

Basic Data Types

- ✓ Stack

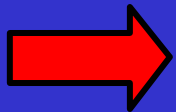
- ✓ Last-In, First-Out (LIFO)

- ✓ initialize, push, pop, status

- ✓ Queue

- ✓ First-In, First-Out (FIFO)

- ✓ initialize, append, serve, status



- List

Overview

- Lists.
- Operations.
- Abstract Data Types (ADT).
- Programming Styles.
- Implementations of Lists.

Lists

APE

0

DEER

1

EMU

2

FOX

3

SEAL

4

Insertion

EMU

Inserted at position 2

Before:



APE

DEER

FOX

SEAL

0

1

2

3

After:

APE

DEER

EMU

FOX

SEAL

0

1

2

3

4

Deletion

Delete item at position 3



Before:



After:



List Operations

- Initialize the list.
- Determine whether the list is empty.
- Determine whether the list is full.
- Find the size of the list.
- Insert an item anywhere in the list.
- Delete an item anywhere in a list.
- Go to a particular position in a list.

A List ADT

A sequence of elements together with these operations:

- Initialize the list.
- Determine whether the list is empty.
- Determine whether the list is full.
- Find the size of the list.
- Insert an item anywhere in the list.
- Delete an item anywhere in a list.
- Go to a particular position in a list.

Abstract Data Type (ADT)

- A Data Structure together with operations defined on it.
- Useful to consider what operations are required before starting implementation.
- Led to the development of object oriented programming.

A Stack ADT

A sequence of elements together with these operations:

- Initialize the stack.
- Determine whether the stack is empty.
- Determine whether the stack is full.
- Push an item onto the top of the stack.
- Pop an item off the top of the stack.

A Queue ADT

A sequence of elements together with these operations:

- Initialize the queue.
- Determine whether the queue is empty.
- Determine whether the queue is full.
- Find the size of the queue.
- Append an item to the rear of the queue.
- Serve an item at the front of the queue.

Comparison

- Stacks
 - Insert at the top of the stack (push)
 - Delete at the top of the stack (pop)
- Queues
 - Insert at the rear of the queue (append)
 - Delete at the front of the queue (serve)
- Lists
 - Insert at any position in the list.
 - Delete at any position in the list.

A Rational Number ADT

Two integers together with these operations:

- Initialize the rational number.
- Get the numerator.
- Get the denominator.
- Simplify a rational number.
- Add two rational numbers.
- Determine whether two rational numbers are equal.
- etc.

A String ADT

A array of characters together with these operations:

- Initialize the string.
- Copy a string.
- Read in a line of input.
- Concatenate two strings.
- Compare two strings.
- Find a length of a string.
- etc.

Programming Styles

- Procedural Programming

Decide which procedures you want: use the best algorithms you can find.

- Modular Programming

Decide which modules you want: partition the program so that data is hidden in modules.

- Data Abstraction

Decide which types you want: provide a full set of operations for each type.

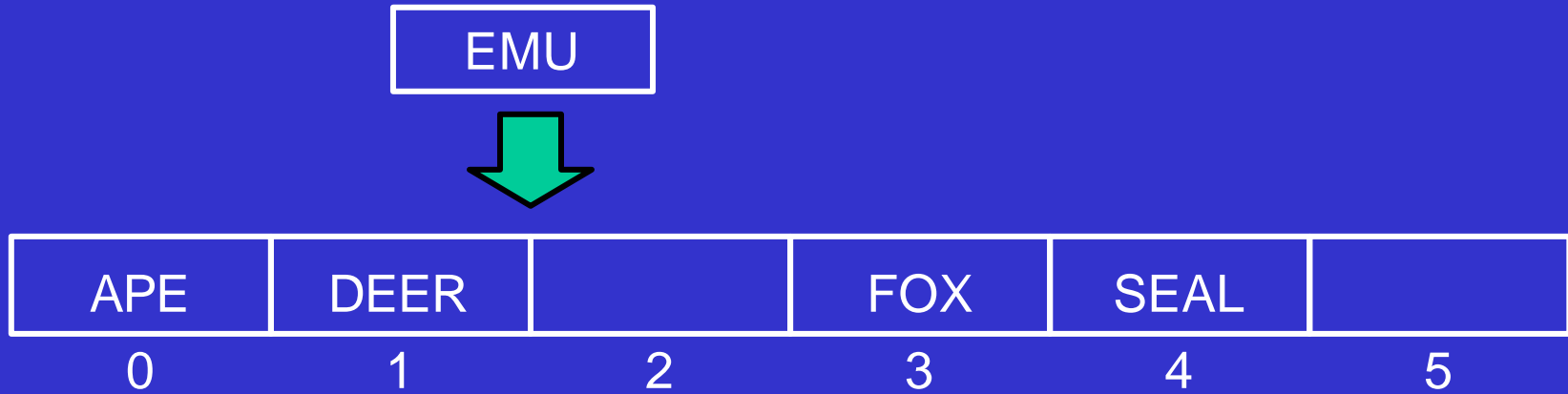
- Object-Oriented Programming

Decide which classes you want: provide a full set of operations for each class; make commonality explicit by using inheritance.

Simple List Implementation



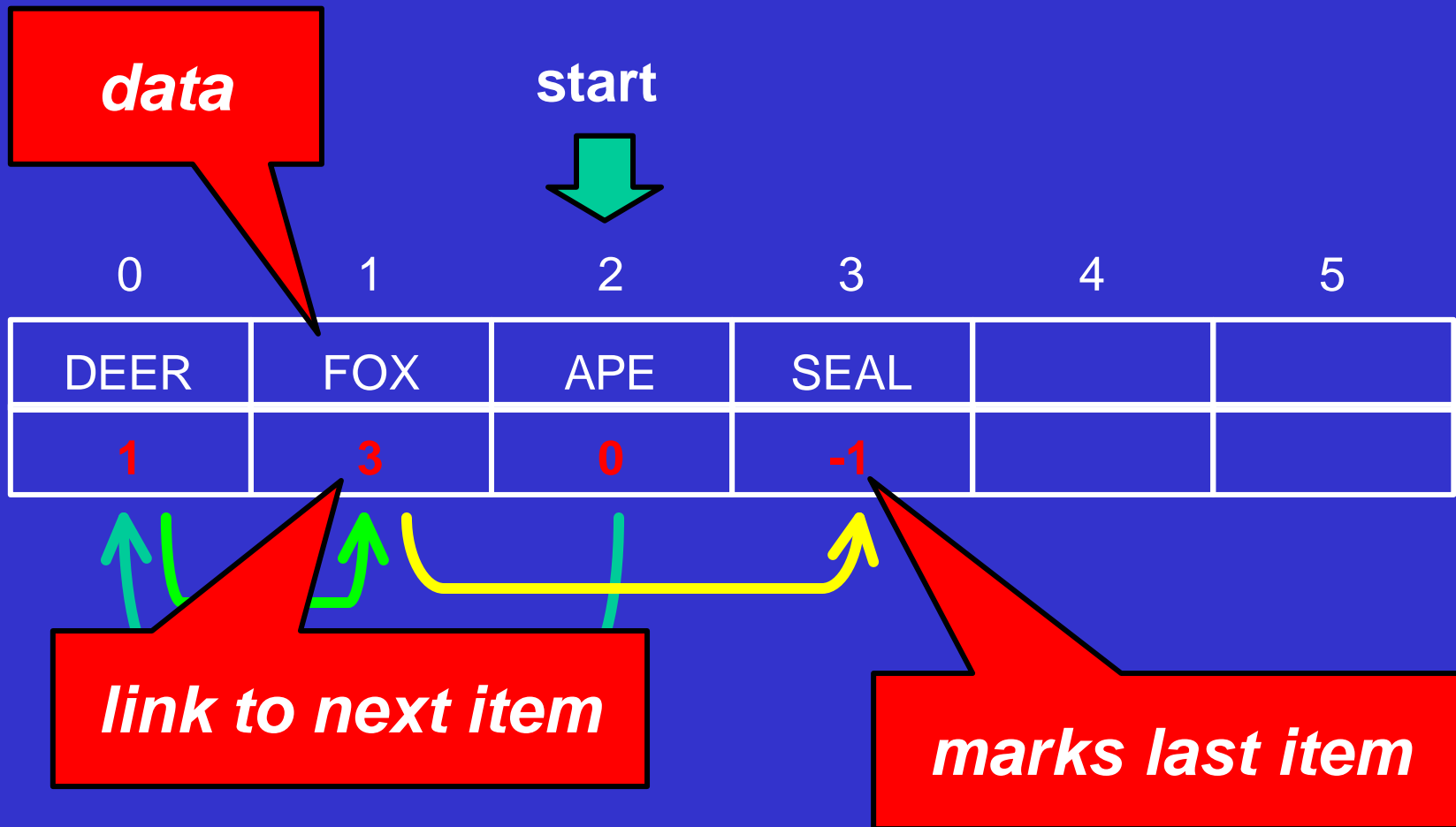
Simple List Implementation



Simple List Implementation

APE	DEER	EMU	FOX	SEAL	
0	1	2	3	4	5

Linked List Implementation: Using Array Index



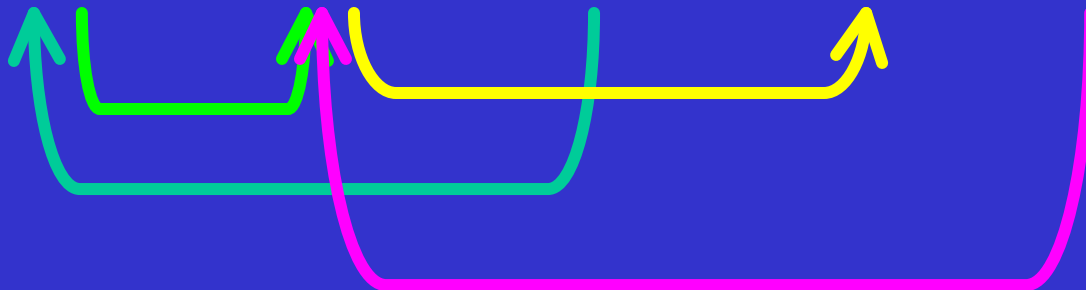
Linked List Implementation: Using Array Index

insert: **EMU**

start



0	1	2	3	4	5
DEER	FOX	APE	SEAL	EMU	
1	3	0	-1	1	



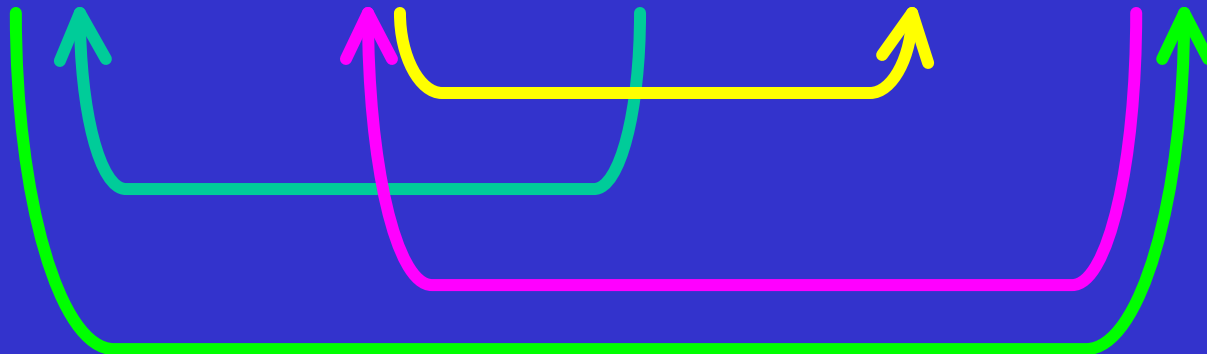
Linked List Implementation: Using Array Index

insert: EMU

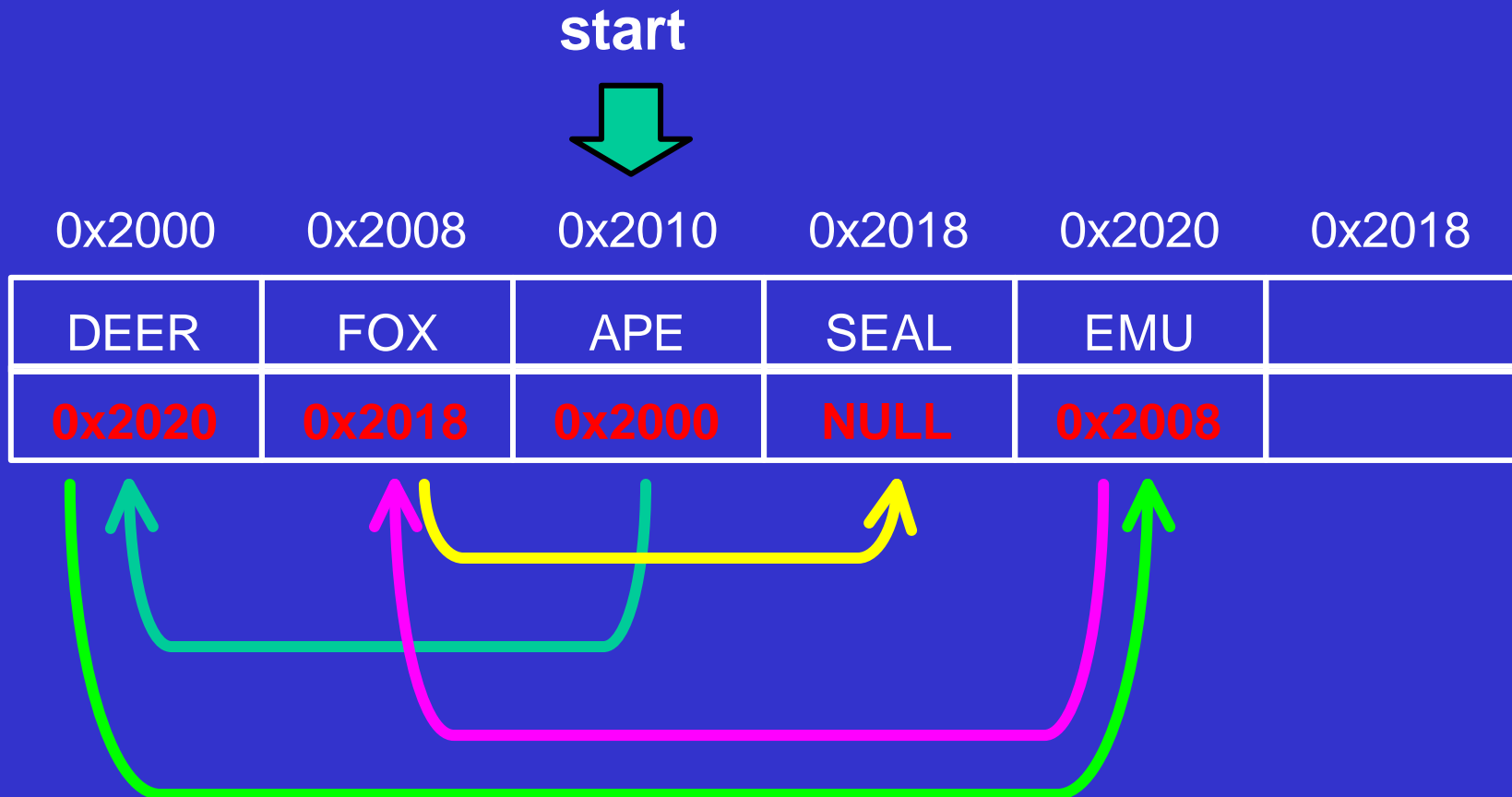
start



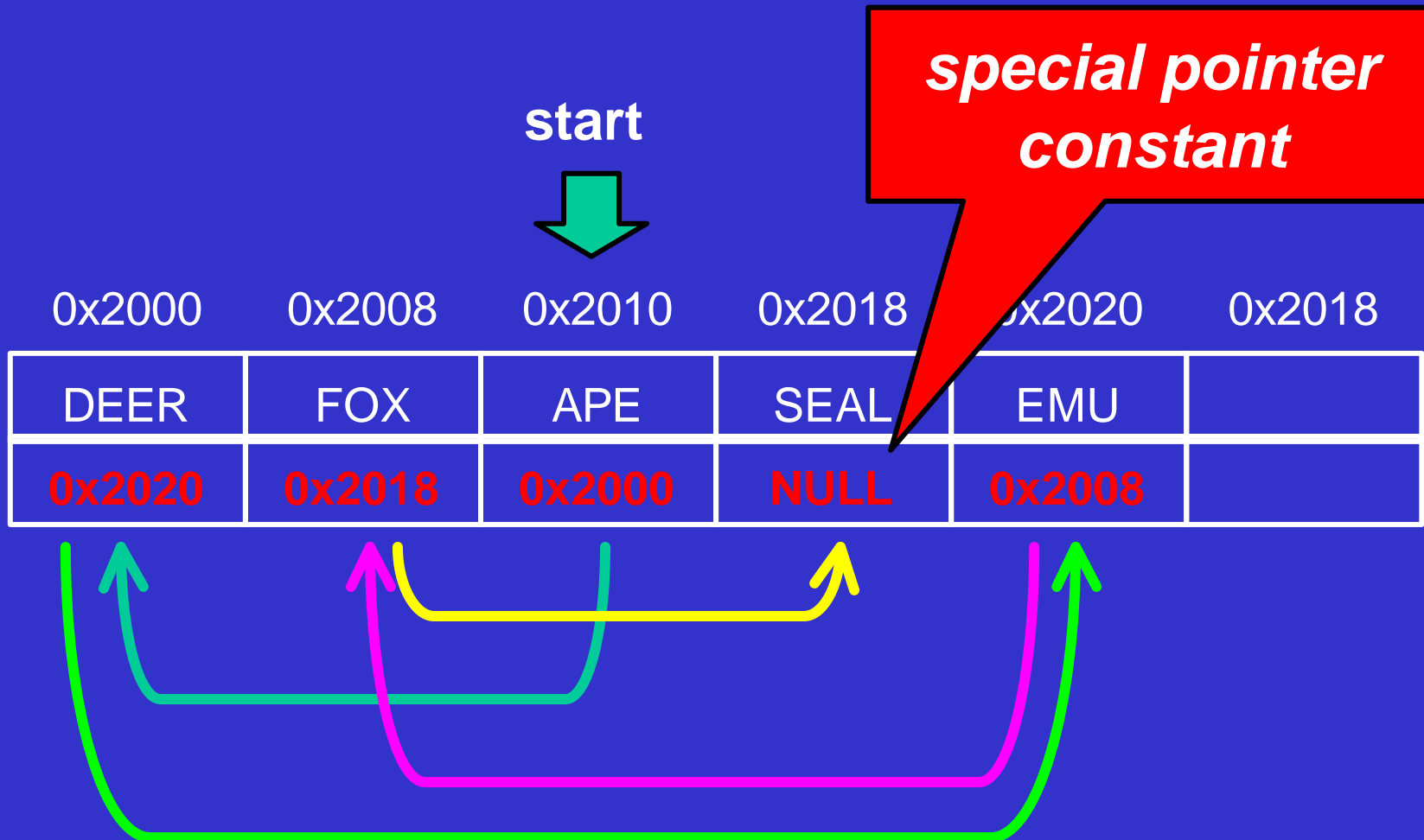
0	1	2	3	4	5
DEER	FOX	APE	SEAL	EMU	
4	3	0	-1	1	



Linked List Implementation: Using Pointers



Linked List Implementation: Using Pointers



Revision

- List
 - initialize, insert, delete, position, status
 - implementations
- Difference between Stacks, Queues, and Lists.
- ADT.

Next Lecture

- Dynamic Memory
- Allocate/Deallocate Memory

Preparation

- Read 4.5 in Kruse et al.
- Read 12.3 in Deitel and Deitel.