CSE 326: Data Structures

Larry Snyder
Autumn Quarter 2006
Lecture 1

CSE 326 Crew

- Paul Pham
- Brian Ngo

Today's Outline

- Introductions
- Administrative Info
- What is this course about?
- Review: Queues and stacks

Course Information

- **Instructor**: Larry snyder, CSE 584
  snyder@cs.washington.edu
- **Text**: *Data Structures & Algorithm Analysis in Java*, 2nd Ed. Mark Allen Weiss, 2007
- **Web page**: `http://www.cs.washington.edu/326`
- **Mailing Lists:**
  - announcement list: *cse326-announce@cs.washington.edu*
  - discussion list: *cse326@cs.washington.edu*

  Subscribe to these using web interface, see homepage

Course Mechanics

- Written homeworks (3-4 total)
  - Due at the **start** of class on due date
  - No late homeworks accepted
- Programming homeworks (3-4 total)
  - In Java
  - Turned in electronically and on paper
  - Once per quarter: use your "late day" for extra 24 hours – **Must email TA in advance**
- Work in teams only on explicit team projects
  - Appropriate *discussions* encouraged – see website

Course Mechanics(2)

- Approximate Grading

  20% - Written Homework Assignments

  25% - Programming Assignments

  20% - Midterm Exam (in class)

  25% - Final Exam (common – different time than listed in UW exam schedule, more coming on this)

  10% - Best of the four items above.

Homework for Today!!

1) **Sign up for mailing lists (immediately)**
2) **Project #1:** (read before section tomorrow)
3) **Preliminary Survey**: fill out by evening of Friday September 29th
4) **Information Sheet**: bring to lecture on Friday, September 29th
5) **Reading** in Weiss (see next slide)

Reading

- Reading in *Data Structures and Algorithm Analysis in Java*, by Weiss
- For this week:
  - › Chapter 1 – (review) Mathematics and Java
  - › Chapter 3 – (Project #1) Lists, Stacks, & Queues
  - › Chapter 2 – (Topic for Friday) Algorithm Analysis

Bring to Class on Friday:

- Name
- Email address
- Year (1,2,3,4)
- Major
- Hometown
- Interesting Fact or what I did over summer/break.

Today's Outline

- Introductions
- Administrative Info
- What is this course about?
- Review: Queues and stacks

Class Overview

Introduction to many of the basic data structures used in computer software
- › Be exposed to a variety of data structures
- › Know when to use them
- › Practice mathematical techniques for analyzing the algorithms that use them
- › Practice implementing and using them by writing programs

Goal:

be able to make good design choices as a developer, project manager, or system customer

Data Structures

"Clever" ways to organize information in order to enable efficient computation

- › What do we mean by clever?
- › What do we mean by efficient?

Picking the best Data Structure for the job

- The data structure you pick needs to *support* the operations you need
- Ideally it supports the operations you will use most often in an *efficient* manner
- Examples of operations:
  - List ADT with operations **`insert`** and **`delete`**
  - Stack ADT with operations **`push`** and **`pop`**

Terminology

- Abstract Data Type (ADT)
  - Mathematical description of an object with set of operations on the object. Useful building block.
- Algorithm
  - A high level, language independent, description of a step-by-step process
- Data structure
  - A specific family of algorithms for implementing an abstract data type.
- Implementation of data structure
  - A specific implementation in a specific language

Terminology examples

- A stack is an *abstract data type* supporting push, pop and isEmpty operations
- A stack *data structure* could use an array, a linked list, or anything that can hold data
- One stack *implementation* is found in java.util.Stack

Concepts vs. Mechanisms

Concepts	Mechanisms
• Abstract	• Concrete
• Pseudocode	• Specific programming language
• Algorithm › A sequence of high-level, language independent operations, which may act upon an abstracted view of data.	• Program › A sequence of operations in a specific programming language, which may act upon real data in the form of numbers, images, sound, etc.
• Abstract Data Type (ADT) › A mathematical description of an object and the set of operations on the object.	• Data structure › A specific way in which a program's data is represented, which reflects the programmer's design choices/goals.

Why So Many Data Structures?

Ideal data structure:

"fast", "elegant", memory efficient

Generates tensions:

- time *vs.* space
- performance *vs.* elegance
- generality *vs.* simplicity
- one operation's performance *vs.* another's

The study of data structures is the study of tradeoffs. That's why we have so many of them!

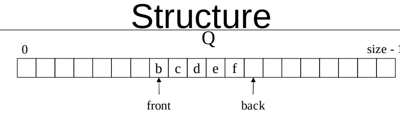
Today's Outline

- Introductions
- Administrative Info
- What is this course about?
- Review: Queues and stacks

First Example: Queue ADT

- Queue operations
  - create
  - destroy
  - enqueue
  - dequeue
  - is_empty

Circular Array Queue Data Structure




```
enqueue(Object x) {
   Q[back] = x ;
   back = (back + 1) % size
   }
dequeue() {
   x = Q[front] ;
   front = (front + 1) % size;
   return x ;
   }
```

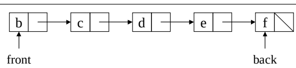
How test for empty list?

How to find K-th element in the queue?

What is complexity of these operations?

Limitations of this structure?

Linked List Queue Data Structure



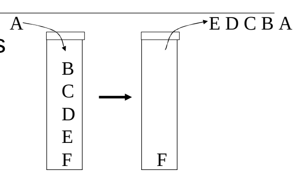

```
void enqueue(Object x) {
  if (is_empty())
      front = back = new Node(x)
  else
      back->next = new Node(x)
      back = back->next
}
bool is_empty() {
  return front == null
}
```

```
Object dequeue() {
  assert(!is_empty)
  return_data = front->data
  temp = front
  front = front->next
  delete temp
  return return_data
}
```


Circular Array vs. Linked List

Second Example: Stack ADT

- Stack operations
  - › create
  - › destroy
  - › push
  - › pop
  - › top
  - › is_empty



Stacks in Practice

- Function call stack
- Removing recursion
- Balancing symbols (parentheses)
- Evaluating Reverse Polish Notation

Homework for Today!!

1) **Sign up for mailing lists (immediately)**
2) **Homework #1:** (read before section tomorrow)
3) **Preliminary Survey**: fill out by evening of Friday September 29th
4) **Information Sheet**: bring to lecture on Friday September 29th
5) **Reading** in Weiss