CSE 326
Lecture 1

Spring Quarter 1995

Course Information

- Instructor: Richard Ladner
  311 Sieg Hall
  543-9347, ladner@cs.washington.edu
  Office Hours: MF 2:30-3:30
  Tu 11:00-12:00
- TA: Dan Fasulo
  326 Sieg Hall
  dfasulo@cs.washington.edu
  Office Hours: M 12:30-1:30 and Th 1:30-2:30
- Text: Data Structures and Algorithm Analysis in C++, by Mark Allen Weiss
  Recommended: C++ Primer, 2nd Edition, by Stanley B. Lippman

Course Policies

- Midterm on Friday, May 5, 1995
- Final on Wednesday, June 7,1995, 2:30
- Written homeworks due every Wednesday
  Homeworks due at the beginning of class and no late homeworks accepted.
- A number of programming projects
  - Projects turned in electonically before 12:01 am.
  - No late projects accepted.
- Grading
  - midterm 20%
  - final 35%
  - homeworks 20%
  - projects 25%

Course Mechanics

- CSE 326 WWW site
- CSE 326 class directory /cse/courses/cse326
- CSE 326 mailing list
- Course Laboratory is 329 Sieg Hall
  - There is no eating or drinking in the Lab

Background Survey

- Computer Science and Enginering Courses

  - CSE 143: Ada or C++

  -CSE 321: Yes or No or current

  -CSE 378: Yes or No or current

  -CSE 341: Yes or No or current

Goals of Course

- Improve ability to think and solve problems abstractly
  - abstraction is a powerful tool in problem solving
- Improve ability to analyze algorithms
  - proof of correctness
  - time complexity analysis
- Become facile with some of the basic data structures used in computer science and engineering
  - at the abstract data type level (ADT)
  - at the algorithm level
  - at the program design level where C++ used as a tool
- Become modestly skilled with the Unix operating system and X-windows. These will be used in future CSE courses.

Plan for this Week

- Monday -
  - course introduction
  - review of concepts needed in course
- Wednesday-
  - continue review of concepts
  - first assignment given
  - introduction to Unix and X-windows
- Thursday
  - review of C++ using examples
- Friday
  - continue introductions to Unix, X-windows and C++
  - first programming assignment given
  - finish review of concepts

Basic Terminology

- Abstract Data Type - ADT

  Mathematical description of an object and set of related operations on the object.

- Algorithm

  A high level, language independent, decription of a step-by-step process for solving a problem.

- Data Structure

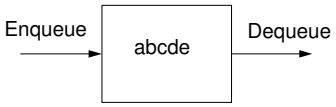
  A family of algorithms for implementing the operations of a given abstract data type. Normally, the term "data structure" refers to abstract data types that are of general use in a diversity of applications. Data structures are the infrastructure of programs, the plumbing and wiring that you don't see.

- Implementation of a data structure

  The computer code that realizes the ADT and its data structure. The code can be written in C++ or other object oriented language.

FIFO Queue ADT

- Queue operation
  - Create
  - Destroy
  - Enqueue
  - Dequeue
  - Is_Empty
  - Is_Full



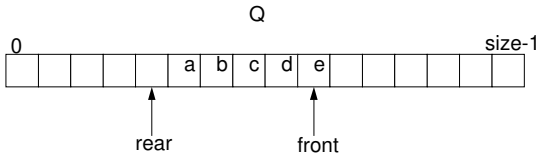
- ADT "axioms" defining the queue (example)

  - If x is Enqueued before y is Enqueued then x will be Dequeued before y will be Dequeued.

  This distinuishes the FIFO Queue ADT from the LIFO Stack ADT where, if x is Pushed before y is Pushed then y will be Popped before x will be Popped.

One Data Structure for a Queue

- The circular array.



- Enqueue(Q,x):

  Q[rear] := x
  rear := (rear - 1) mod size;

- Dequeue(Q)

  x := Q[front];
  front := (front -1) mod size;
  return(x)

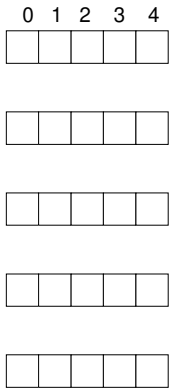
- Is_Empty(Q)

  return (front = rear)

- Is_Full(Q)

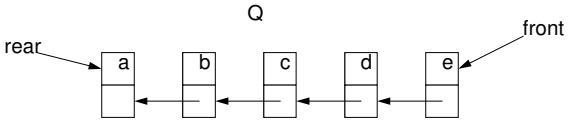
  return (front = (rear - 1) mod size)

Queue Example



Another Data Structure for a Queue

- Linked list



- Enqueue(Q,x)

  p := newnode;
  p.data := x;
  p.next:= nil;
  rear.next := p;
  rear := p

- Dequeue(Q)

  x := front.data;
  front := front.next;
  return(x)