Data Structures and Algorithms CSCI 2270

Rhonda Hoenigman
January 17, 2018
Lecture 1

University of Colorado Boulder

Today…

- 2270 details
  - Grading
  - Textbook
  - Recitations
  - Computer Science Moodle
  - A little bit about me, and computer science

What is Data Structures and Algorithms?

Back to Intro Programming…

Most of you are coming from CSCI1300, 1320

Reasons for enrolling in CSCI1300

- Intentional interest
- Required for another major
- Your mom made you do it
- You got lost on your way to English 100
- Accidental

If you are here, presumably you enjoy programming and want to know more.

Administrative details

Course materials on Computer Science Moodle
http://moodle.cs.colorado.edu/
CSCI 2270 – Hoenigman/Zagrodzki/Zietz – CS2: Data Structures
Enrollment key: csci2270

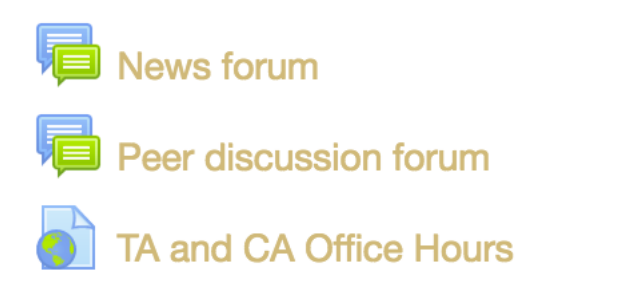
Login with your identikey and password

Class times:
MWF, 3-3:50pm. MATH100
Recitation:
Weekly, 1.25-hour meetings.
Recitation activity. Ask questions about assignments and get extra help.

Administrative details

Undergrad Course Assistants (CA):

Full schedule on Moodle – times and locations



My Office hours:
W, F, 12:00-1:00pm., Th, 12:30-1:30pm. ECOT 738

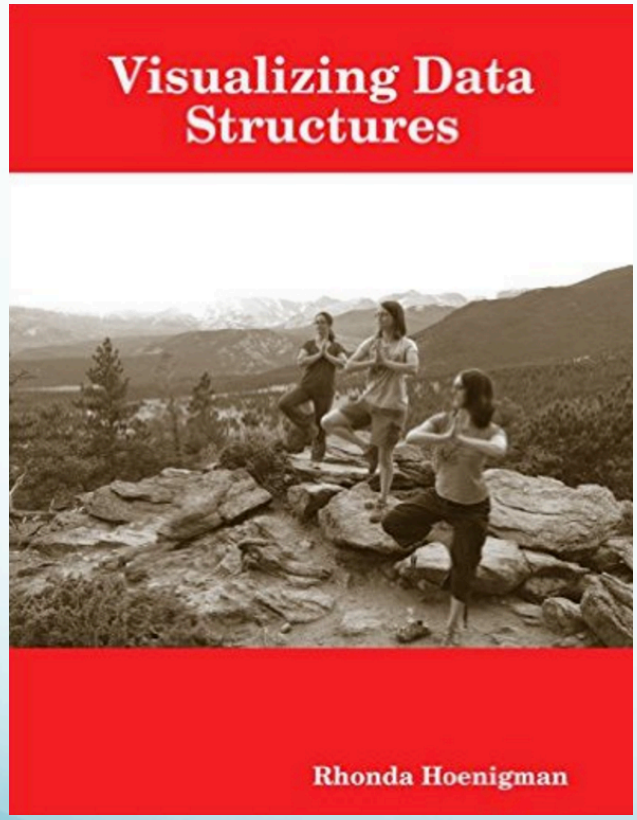
Office hour information is also in the syllabus.

Programming pre-requisite for CSCI 2270

- You should be proficient in a programming language, preferably C++
- Learning a new language is part of computer science
- Resources for learning C++
  - My lecture notes from CSCI1300, posted on Moodle
  - On-line C++ textbook
  - Bucky's C++ video tutorials

Course resources

- Required ebook: Visualizing Data Structures, Hoenigman, 2015.



Available on Moodle for free, or Amazon, Google, iTunes for about $10.

Lecture format

- **Bring your laptop to class everyday**
- **Bring paper and pencil, or some way of taking notes, to class everyday**
- Class is theory and implementation
  - Pseudo-code for algorithms
  - Implement algorithms in lecture

Grading policy

Recitation 15%
- Weekly assignments

Assignments 45%
- Weekly programming assignments

Project 10%
- End of semester

Lecture quizzes 10%
- 2-3 questions after every lecture. Due before next lecture.

Midterms (two midterms, evening exams) 20%
- Must have a 65% average on the exams to get better than a D+ in the class, regardless of other grades.

Final exam – replace your lowest midterm

Recitation this week

Get prepared for the semester: Sublime, Moodle, C++

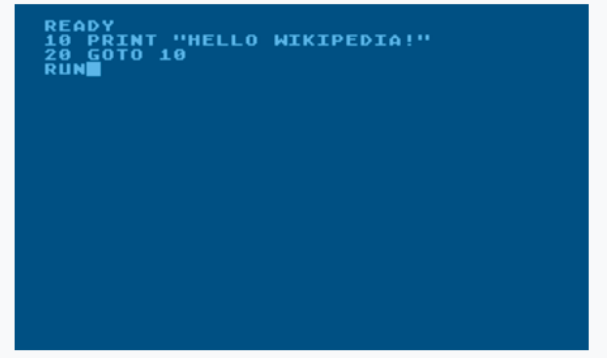
If you used the Virtual Machine in previous semester, you're welcome to continue with it.

Please read the syllabus.

Create an account on the Moodle
http://moodle.cs.colorado.edu/
Enrollment key: csci2270

Programming exercise:
Write a C++ function to determine if a number is prime.

Computing – it's always changing



Atari BASIC

Commodore BASIC v2.0 on the Commodore 64

Computing – 20+ years later

```
class Rectangle {
    int width, height;
  public:
    void set_values (int,int);
    int area (void);
} rect;
```

C++ class

```
class Customer(object):
    """A customer of ABC Bank with a checking account. Customers have the
    following properties:

    Attributes:
        name: A string representing the customer's name.
        balance: A float tracking the current balance of the customer's accou
    """

    def __init__(self, name, balance=0.0):
        """Return a Customer object whose name is *name* and starting
        balance is *balance*."""
        self.name = name
        self.balance = balance
```

Python

Processing power
Languages
Language design
New versions of languages

My path through computing

- BASIC
- Assembly
- C/C++
- HTML, ASP, Javascript, PHP
- SQL
- Java
- Visual Basic, C#
- Python
- Matlab

Education and work:

- BS Journalism, 1994
- MS Computer Science, 2007
- PhD Computer Science, 2012

- 10 Years experience in software engineering, 1997-2007

Skills for success

- Willingness to learn new things, adaptability
- Motivation
- Curiosity
- Self-study – learn new languages, technologies
- Don't have to have the right answer
- Practice

1993. Petersburg, AK annual July 4 log-rolling contest. Did I win?

Stick to computing

No, I didn't win.

Common to all - Data Structures and Algorithms

Catalog Definition:

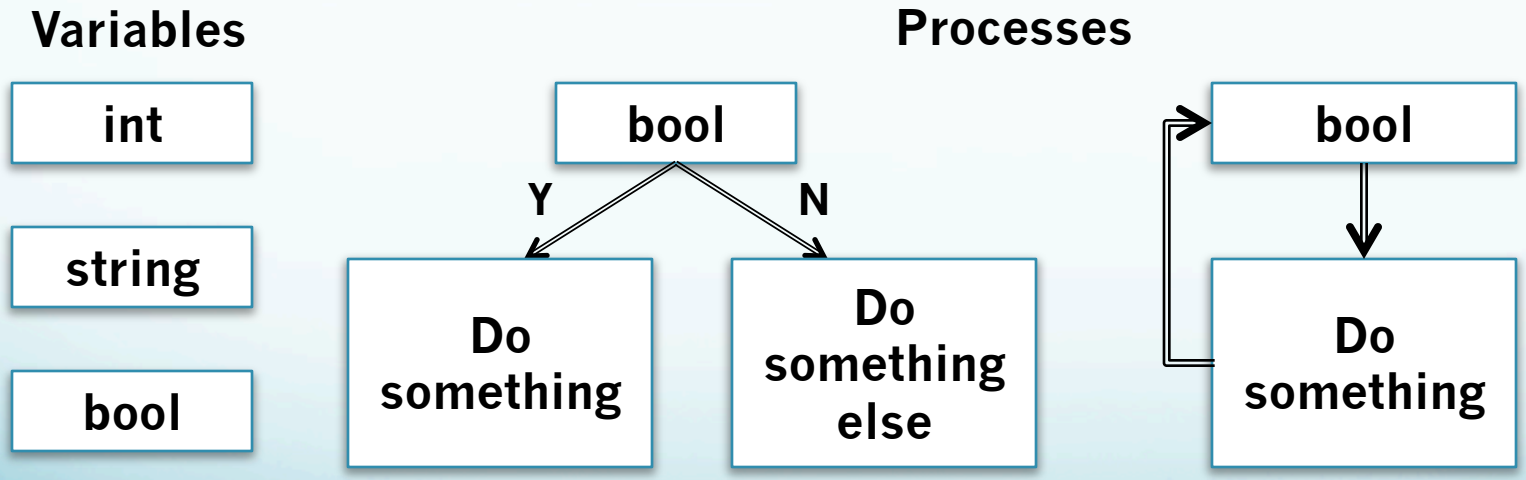
- Studies data abstractions (e.g., stacks, queues, lists, trees) and their representation techniques (e.g., linking, arrays). Introduces concepts used in algorithm design and analysis including criteria for selecting data structures to fit their applications.

- Wow, really? That's a lot of words.

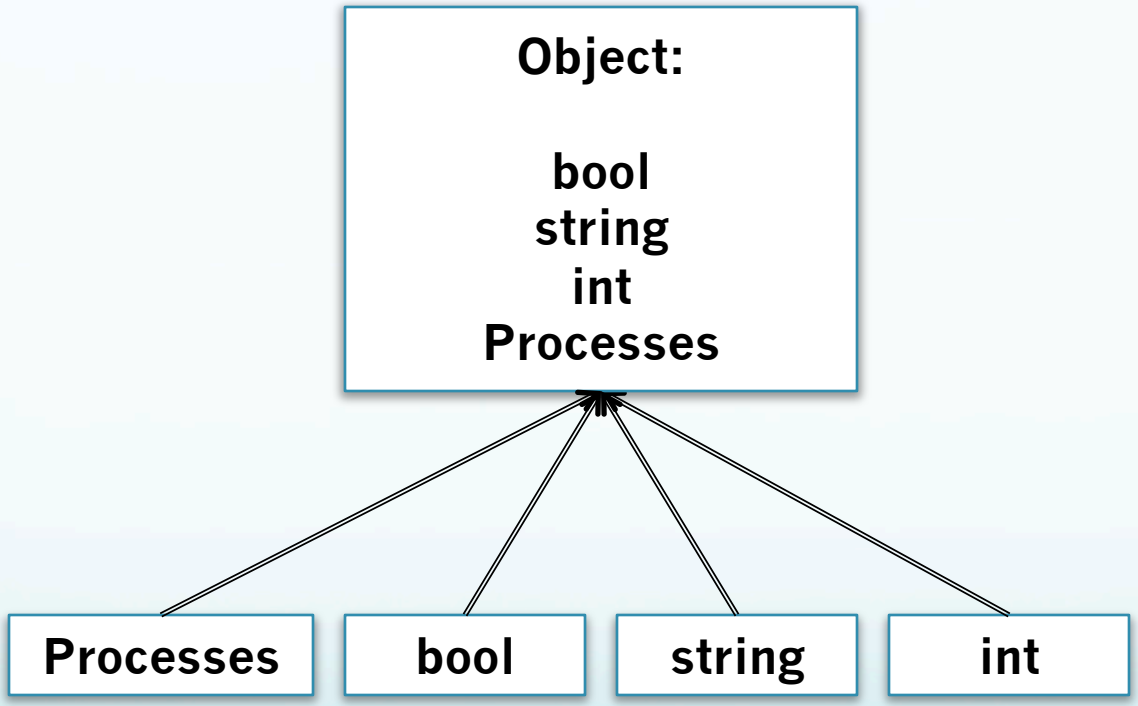
- **First part:** Studies data abstractions (e.g., stacks, queues, lists, trees) and their representation techniques (e.g., linking, arrays).

Data abstractions

- Programming:
  - Computational representation of the world
  - Abstract meaningful details for representation
  - Variables and processes



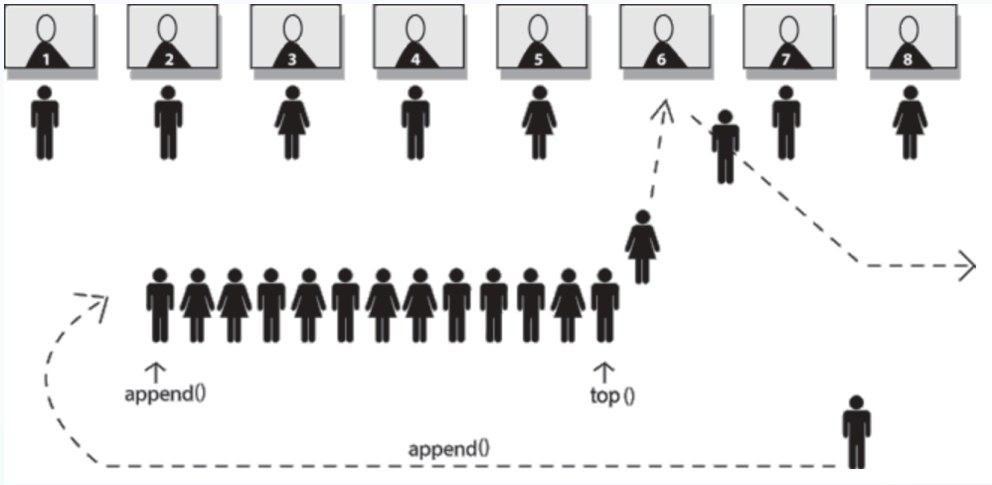
Data abstractions - objects



Data abstractions - collections of objects

Object: Person

One person

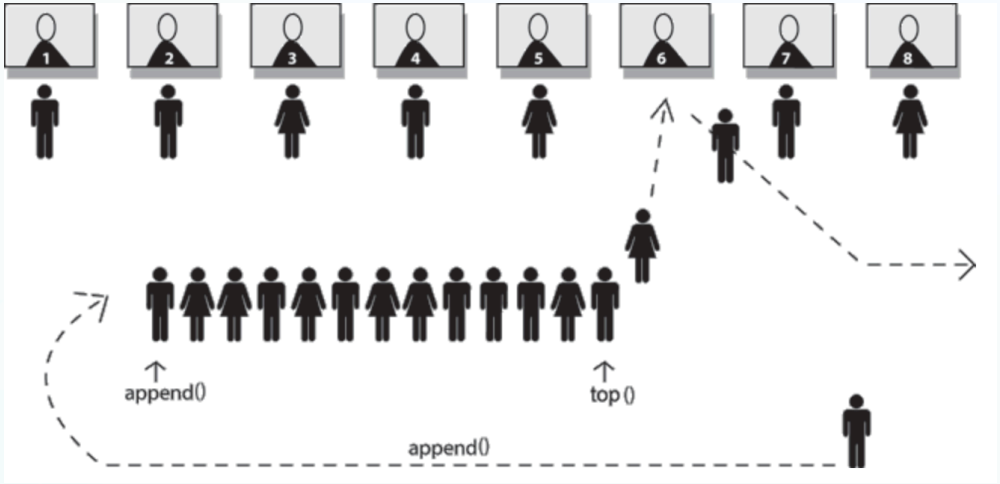


Collection of persons in line at a store.

Features:

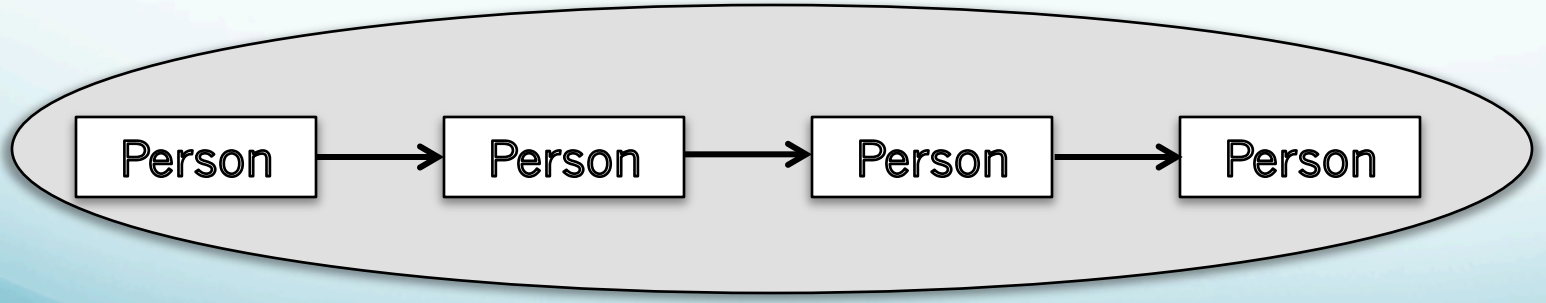
- Individual person objects
- Order
- Adding at one end, remove at the other

Queue data structure

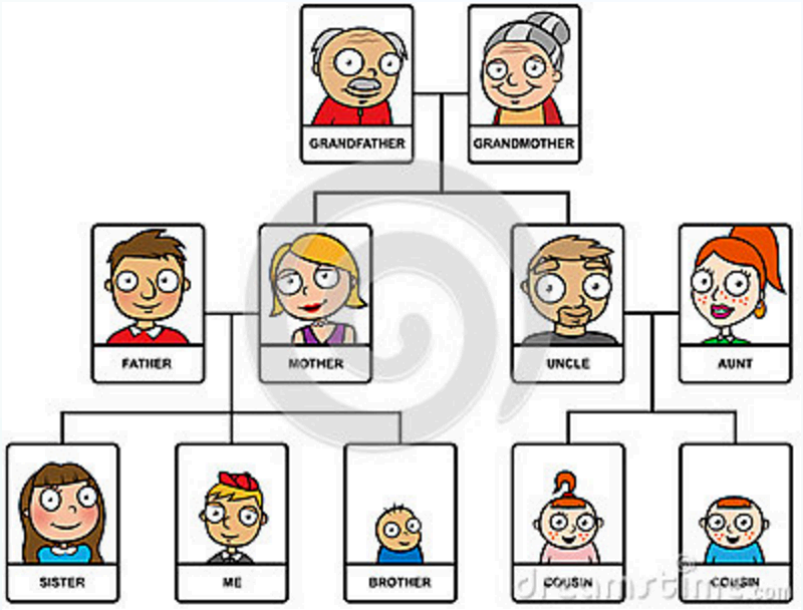


Example:
Processes scheduled by an operating system.

Queue – add at one end, remove from the other



Family tree



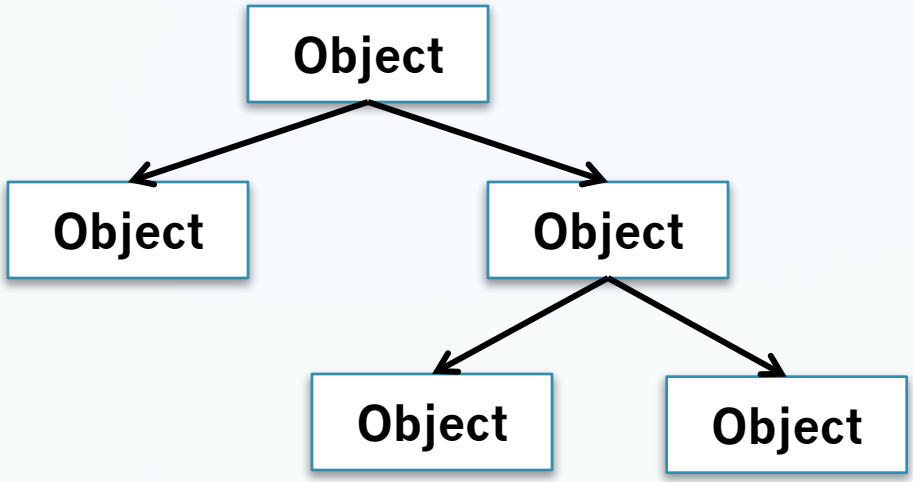
Object: Person

Collection of persons in a hierarchy

Family tree features

- Individual persons
- Parent-child person ordering

Tree data structure



Example:
Binary search tree for sorting

Collection of objects in a hierarchy

Tree features

- Individual objects
- Parent-child object ordering
- Adding, removing, and maintaining tree structure
- Searching

Arrays and stacks

An array of egg data

Array features:

- Contiguous in memory
- Fixed locations
- Egg[0] next to Egg[1]
- Add, remove, search

An stack of plate data

Stack features:

- Add and remove from the top
- Order matters

Example:
Commands executed in computer program

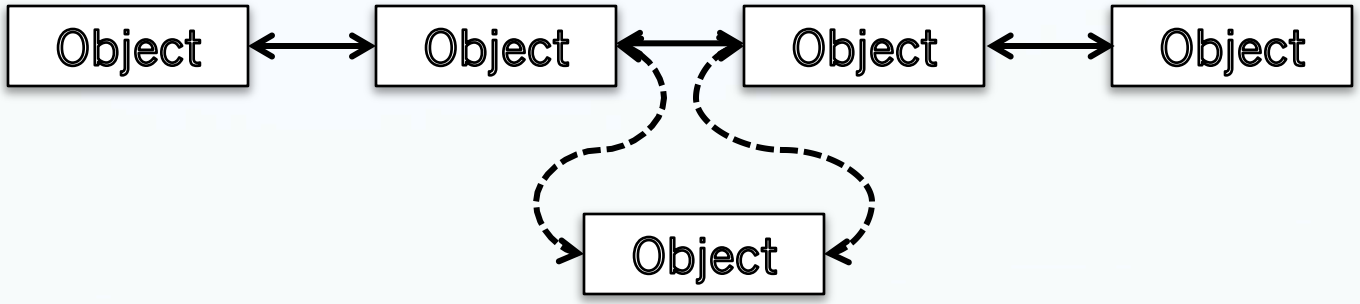
Towers linked together

Collection of objects linked together

Features:

- Individual tower objects
- Wires between towers establish order
- Add, remove by changing wires
- Locations not fixed
- Number not fixed

Linked list



Collection of objects linked together

Features:

- Individual tower objects
- Pointers in memory establish order
- Add, remove by changing pointers
- Locations not fixed
- Number not fixed

Example:
Dynamic data storage

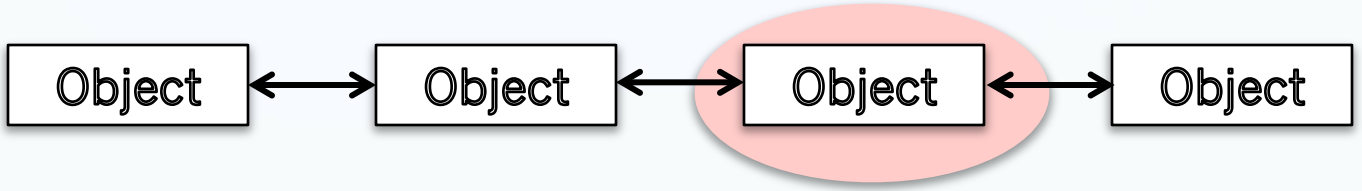
Data Structures and Algorithms

Catalog Definition:

- Studies data abstractions (e.g., stacks, queues, lists, trees) and their representation techniques (e.g., linking, arrays). **Introduces concepts used in algorithm design and analysis including criteria for selecting data structures to fit their applications.**

Algorithm design and analysis

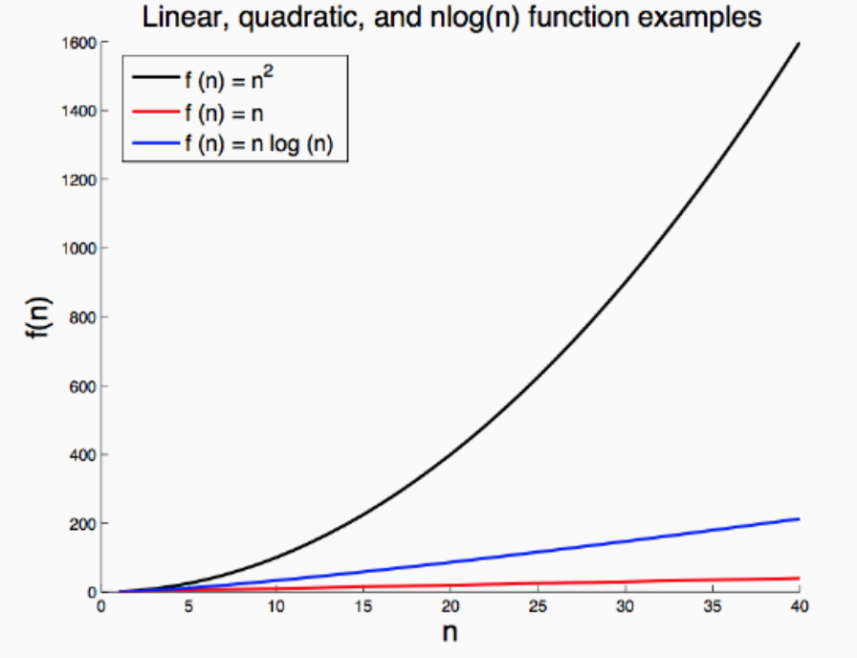
- How many operations to access a linked list?



- Operations on an array?

Algorithm design and analysis

- How does an algorithm scale as data becomes really large?



- n is the size of the data structure, e.g. number of elements in an array.
- f(n) is an operation on the data structure of size n.

What you will learn in this class

- How to build data structures
  - E.g.: Arrays, linked lists, stacks, queues, trees, graphs, hash tables
- Why one data structure is better than another for a certain problem.
- Complexity of operations on data structures
  - Search
  - Insert
  - Delete

This week

- Assignment 1 posted on Moodle, due Jan 26.
- Recitation 1 due Sunday
- Friday lecture: C++ review