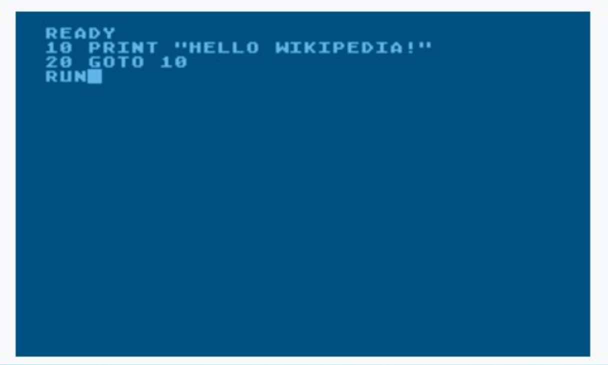
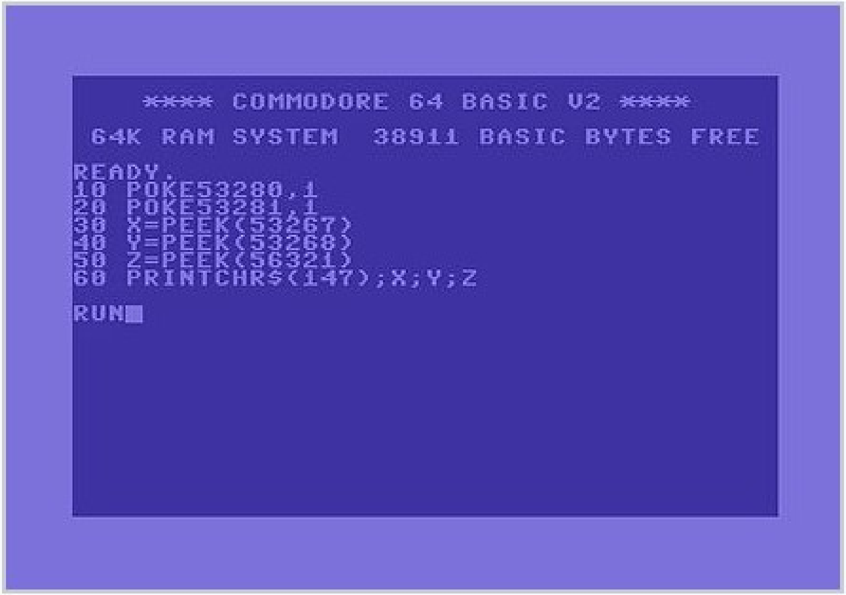
Computing – it's always changing



```
READY
10 PRINT "HELLO WIKIPEDIA!"
20 GOTO 10
RUN█
```

Atari BASIC

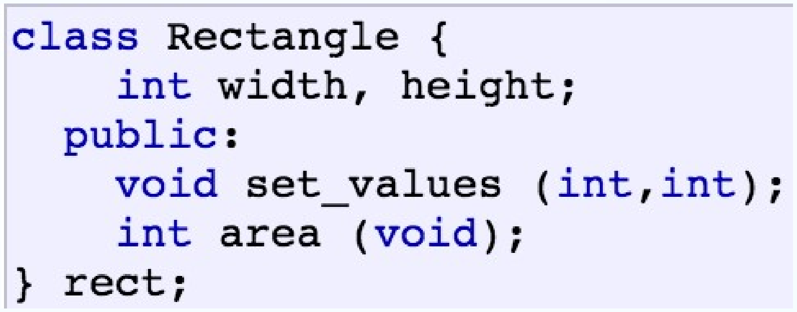


```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
10 POKE53280,1
20 POKE53281,1
30 X=PEEK(53267)
40 Y=PEEK(53268)
50 Z=PEEK(56321)
60 PRINTCHR$(147);X;Y;Z

RUN█
```

Commodore BASIC v2.0 on the Commodore 64

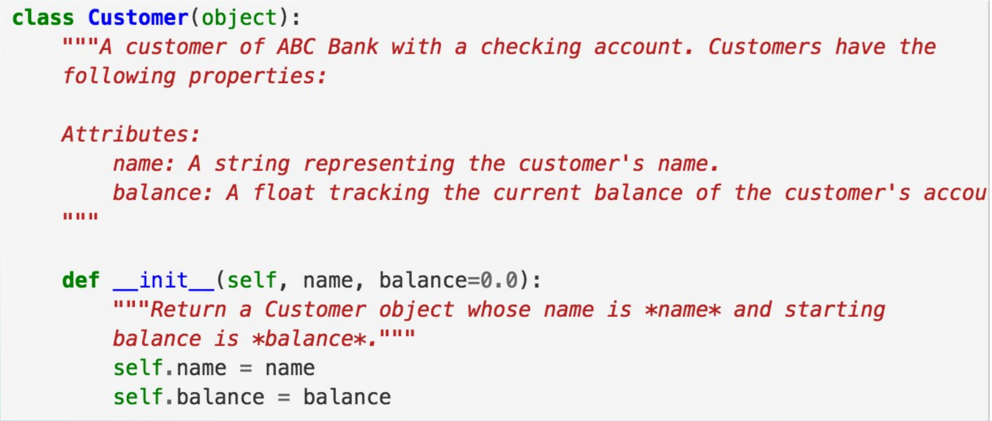
University of Colorado Boulder

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

Common to all - Data Structures and Algorithms

Catalog Definition:

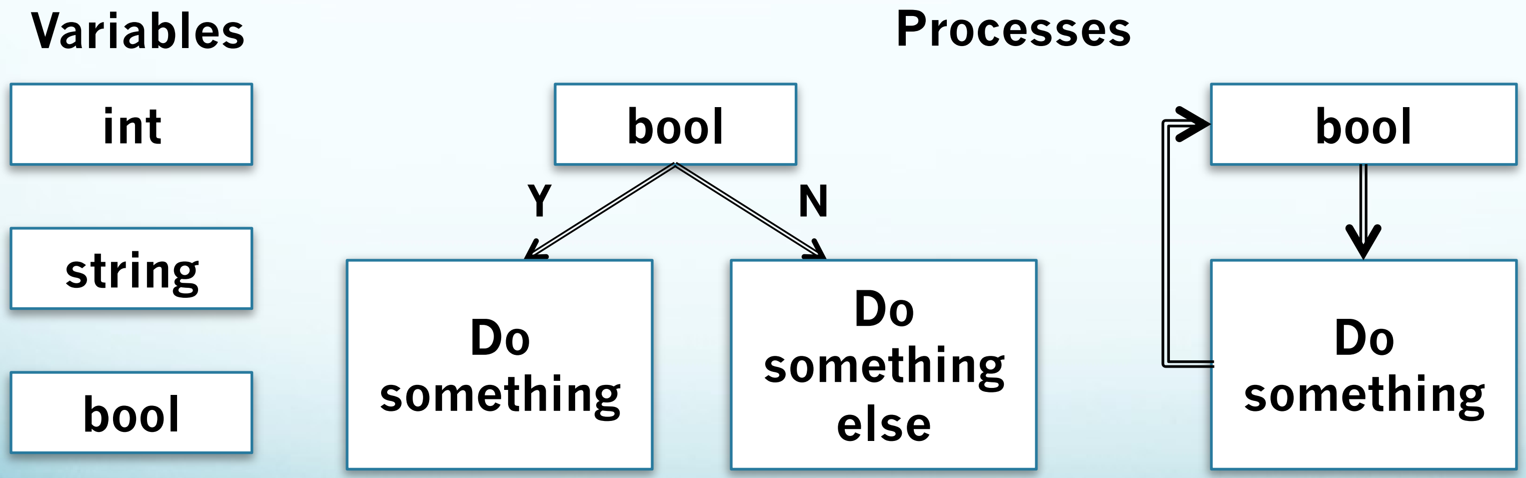
- Studies data abstractions (e.g., stacks, queues, lists, trees) and their representation techniques (e.g., linking, arrays). Introduces concepts used in algorithm design and analysis including criteria for selecting data structures to fit their applications.

- 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



Variables

int

string

bool

Processes

bool

Y

N

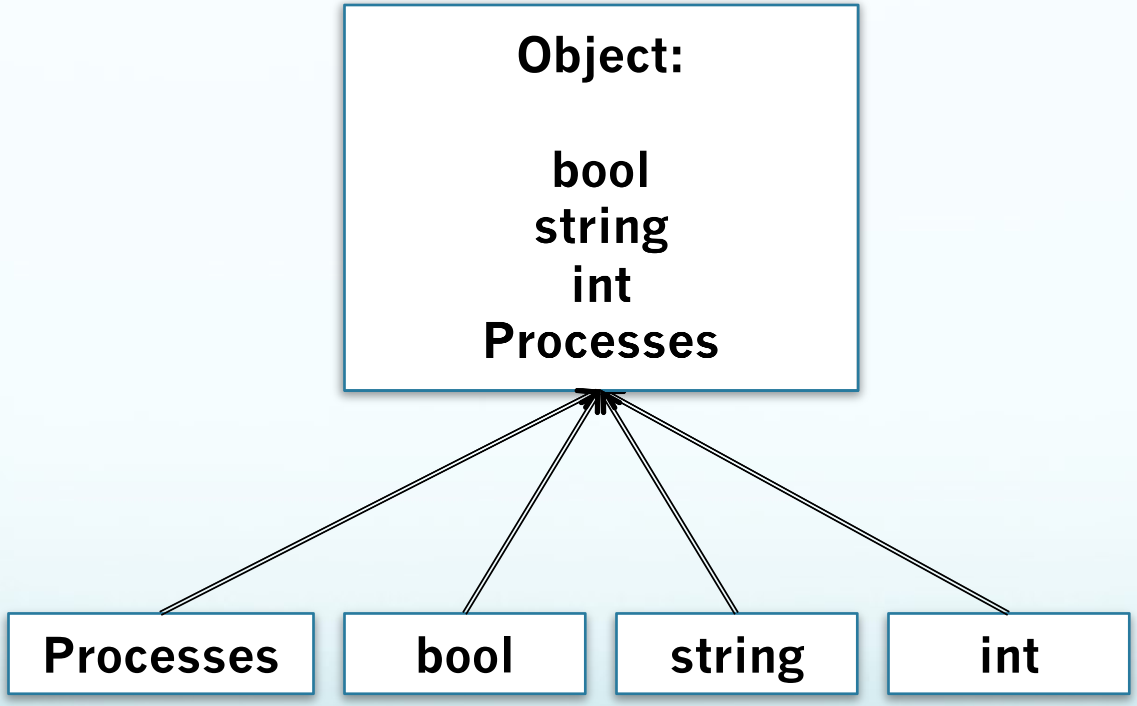
Do something

Do something else

bool

Do something

Data abstractions - objects



Object:

bool
string
int
Processes

Processes

bool

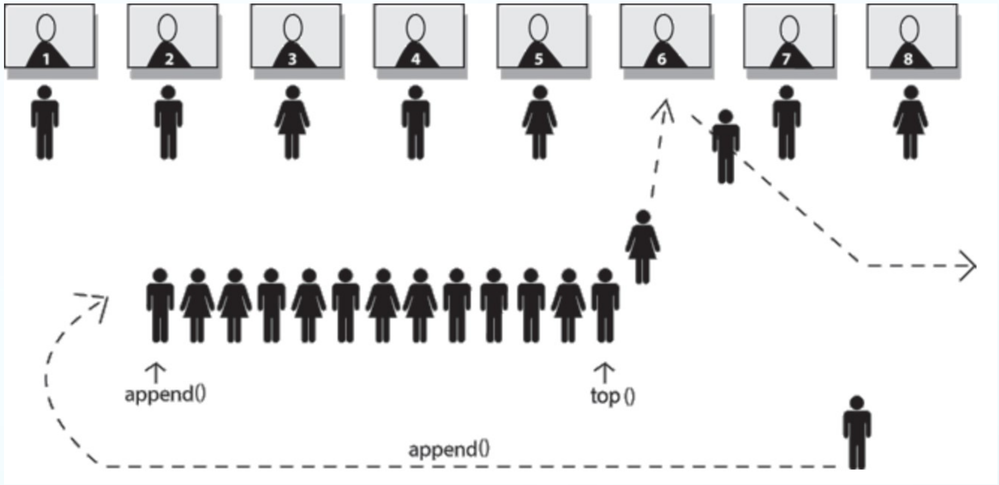
string

int

Data abstractions - collections of objects

Object: Person

One person

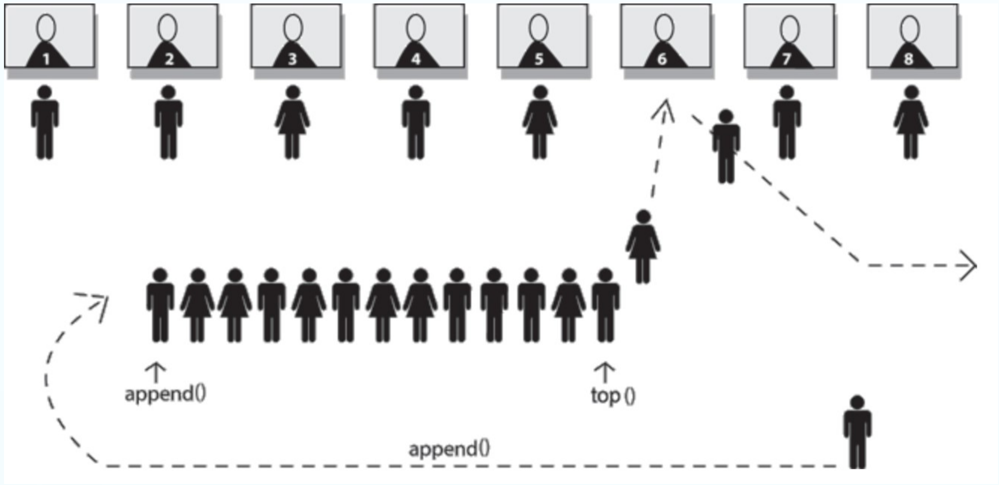


Collection of persons in line at a store.

Features:

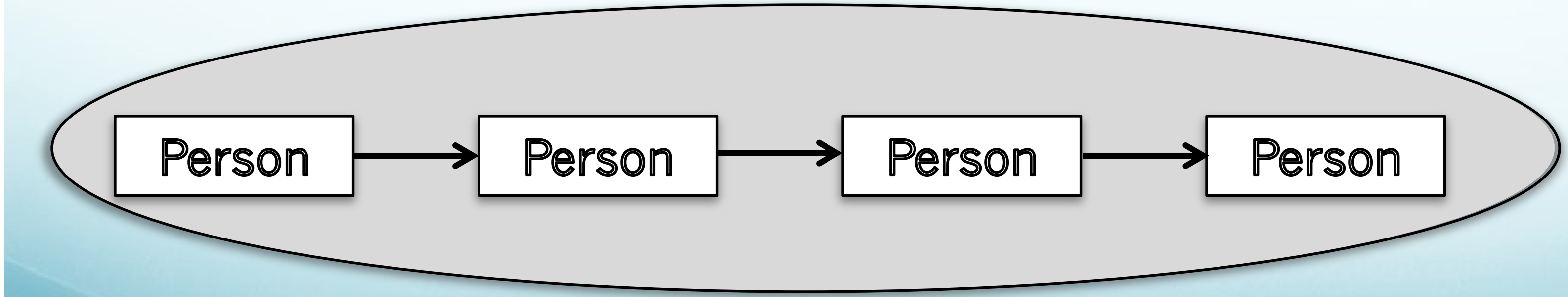
- Individual person objects
- Order
- Adding at one end, remove at the other

Queue data structure

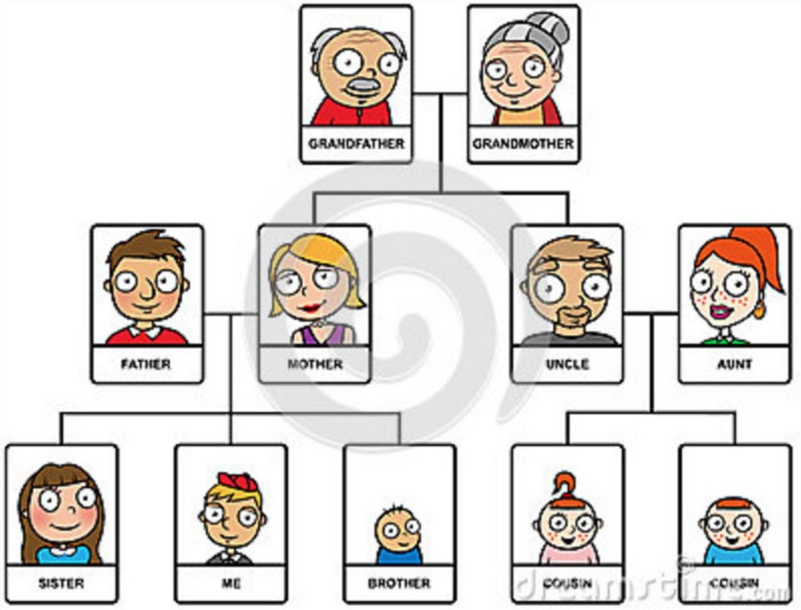


Example:
Processes scheduled by an operating system.

Queue – add at one end, remove from the other



Family tree



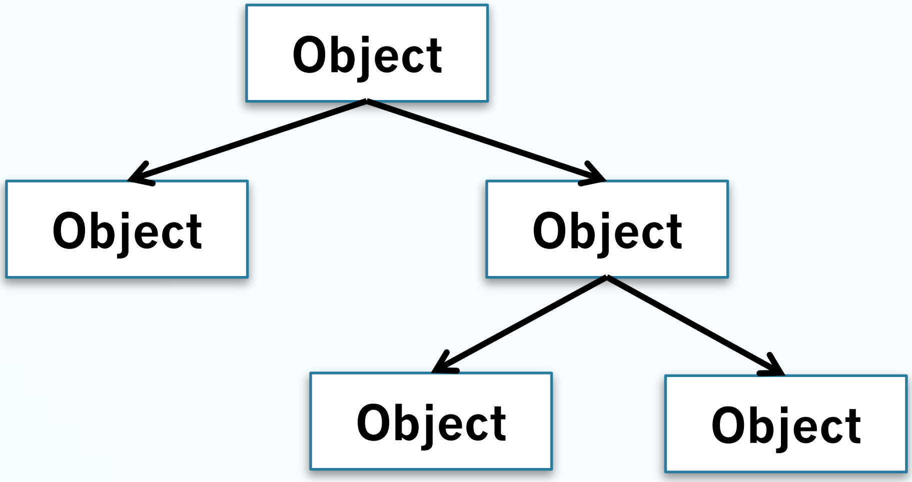
Object: Person

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

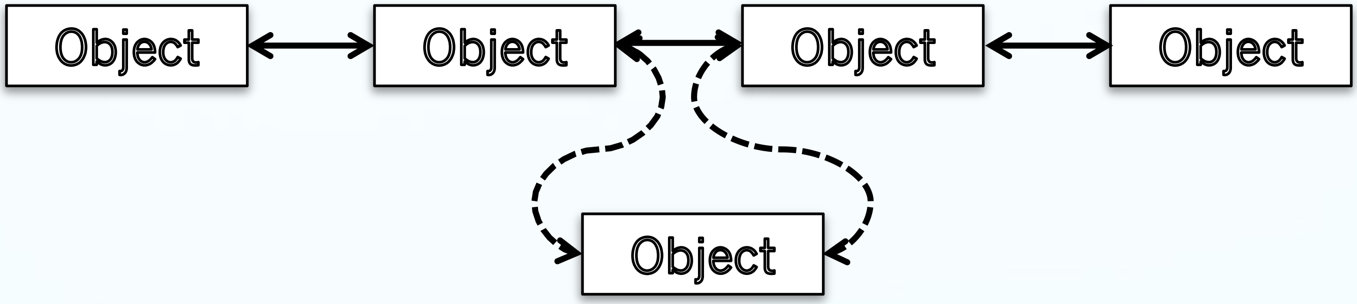
Towers linked together

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

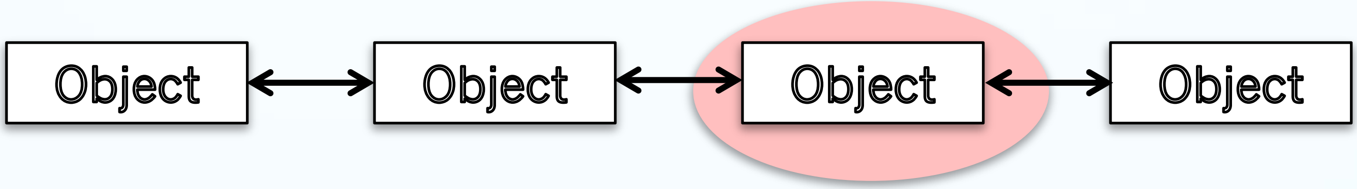
Data Structures and Algorithms

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?



Object

Object

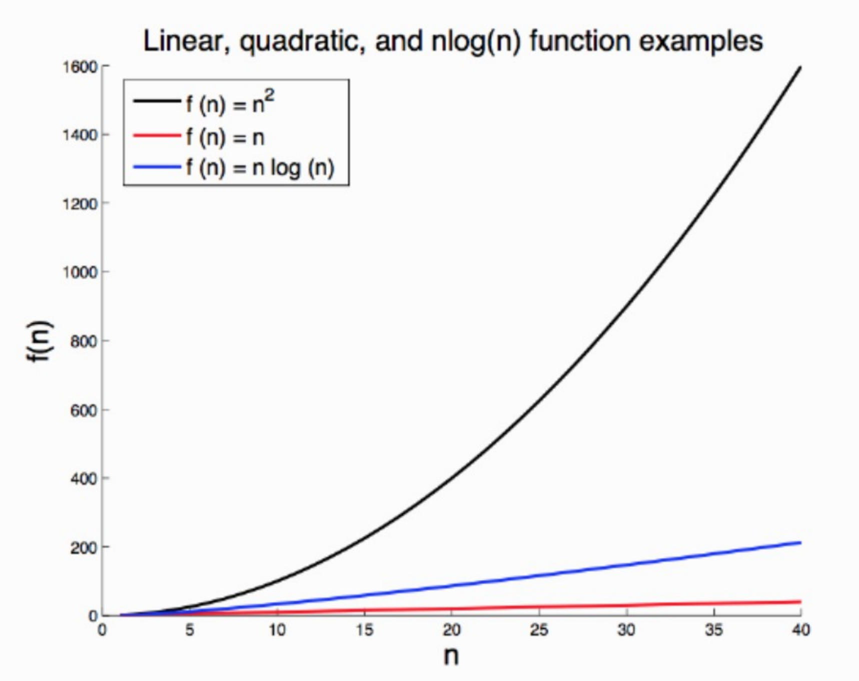
Object

Object

- 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.

University of Colorado
Boulder

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