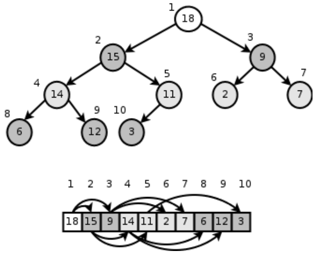
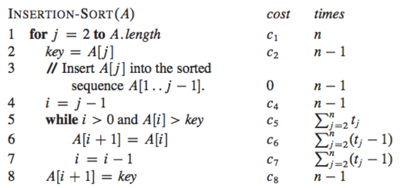
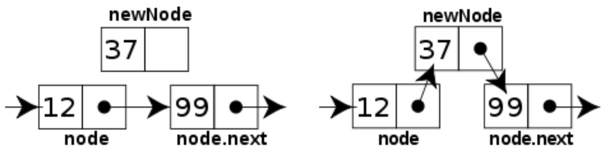




```
INSERTION-SORT(A)                                   cost   times
1  for j = 2 to A.length                            c_1    n
2      key = A[j]                                   c_2    n − 1
3      // Insert A[j] into the sorted
           sequence A[1 . . j − 1].                 0      n − 1
4      i = j − 1                                    c_4    n − 1
5      while i > 0 and A[i] > key                   c_5    Σ_{j=2}^{n} t_j
6          A[i + 1] = A[i]                          c_6    Σ_{j=2}^{n} (t_j − 1)
7          i = i − 1                                c_7    Σ_{j=2}^{n} (t_j − 1)
8      A[i + 1] = key                               c_8    n − 1
```



WELCOME TO CS 24!

Problem Solving with Computers-II

Read the syllabus. Know what's required. Know how to get help.

Enrollment status:
120/120
Waitlist: 23

About me

- Diba Mirza (diba@ucsb.edu)
  - Assistant Teaching Professor, Computer Science
  - PhD (Computer Engineering, UCSD)

- Office hours (starting next week 04/08):
  - W: 11a -noon, F: noon – 1p Or by appointment
  - Location: HFH 1155
- You can reach me via
  - Piazza (highly recommended)
  - Email: Include [CS24] on the subject line

Course staff

TAs

Dheeraj

Mohith

Lead tutors

Madhu

Arthur

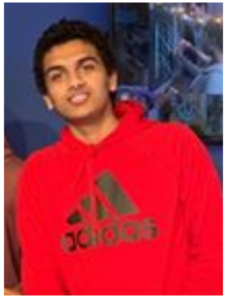
TAs and tutors:

- One-one help in during "closed"
  "open labs"
- Open labs start from week 2
- Feedback on code
- Any question related to CS, internships, courses, UCSB...

Eliza

Jared

Lily


Adarsha

Iris

Tina

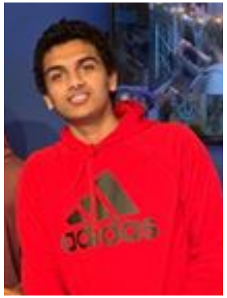
Aesha

Course staff- Learn their names!

TAs

Lead tutors

Open labs will start from week 2



Getting help

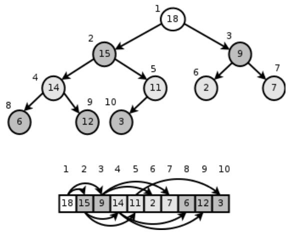
- Come to office hours and introduce yourself
- Setup a regular time to meet outside of section time with TAs and tutors
- Communicate with the staff in person and remotely on:

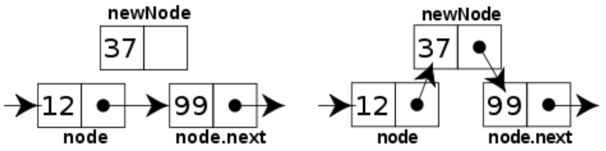
PIAZZA!

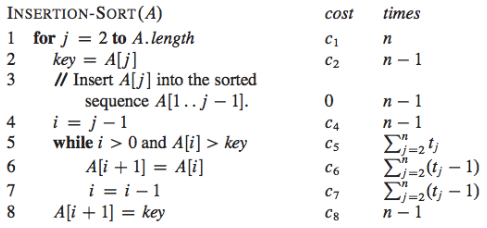
About this course

You will learn to:

- Design and implement **larger programs** that **run fast**
- Organize **data** in programs using **data structures**
- **Analyze** the **complexity** of your programs







```
INSERTION-SORT(A)                                      cost    times
1  for $j = 2$ to $A.length$                           $c_1$   $n$
2      $key = A[j]$                                    $c_2$   $n-1$
3      // Insert $A[j]$ into the sorted
           sequence $A[1..j-1]$.                       0       $n-1$
4      $i = j - 1$                                     $c_4$   $n-1$
5      while $i > 0$ and $A[i] > key$                  $c_5$   $\sum_{j=2}^{n} t_j$
6          $A[i+1] = A[i]$                             $c_6$   $\sum_{j=2}^{n} (t_j - 1)$
7          $i = i - 1$                                 $c_7$   $\sum_{j=2}^{n} (t_j - 1)$
8      $A[i+1] = key$                                  $c_8$   $n-1$
```

Data Structures and C++ **Complexity Analysis**

Course Logistics

- Coure website: https://ucsb-cs24.github.io/s19/
- Grading
    - Homeworks: 10%
    - Lab assignments: 15%
    - Programming assignments: 20%
    - Midterm Exams: 25%
    - Final Examination: 30%

- NO MAKEUPS ON EXAMS!

- You have 24 hour grace period to submit the labs. DO NOT contact the instructor or TAs for extensions unless you have a real emergency

- To complete the labs you need a college of engineering account. If you don't have one yet, send an email to help@engineering.ucsb.edu

iClickers: You must bring them

- Buy an iClicker at the Bookstore
- Bring your iclicker to class

Required textbook

- Michael Main and Walter Savitch. *Data Structures and Other Objects Using C++ (4th edition)*, Addison-Wesley, 2011.

Recommended textbook

- Problem Solving with C++, Walter Savitch, Edition 9

You must attend class and lab sections
You must prepare for class
You must participate in class

Clickers out – frequency AC

About you…

What is your familiarity/confidence with programming in C++?

A. Know nothing or almost nothing about it.
B. Used it a little, beginner level.
C. Some expertise, lots of gaps though.
D. Lots of expertise, a few gaps.
E. Know too much; I have no life.

About you…

What is your familiarity/confidence with using version control – git or subversion?

A. Know nothing or almost nothing about it.
B. Used it a little, beginner level.
C. Some expertise, lots of gaps though.
D. Lots of expertise, a few gaps.
E. Know too much; I have no life.

About you…

What is your familiarity/confidence with C++ memory-management?

A. Know nothing or almost nothing about it.

B. Used it a little, beginner level.

C. Some expertise, lots of gaps though.

D. Lots of expertise, a few gaps.

E. Know too much; I have no life.

Clickers, Peer Instruction, and PI Groups

- Find 1-2 students sitting near you. If you don't have any move.
- Introduce yourself.
- This is your initial PI group (at least for today)
- Discuss what you hope to get out of this class.

Procedural Programming

- Break down a problem into sub tasks (functions)
- Algorithm to bake a cake

    Preheat the oven to 350F

    Get the ingredients: 2 eggs, 1 cup flour, 1 cup milk

    Mix ingredients in a bowl

    Pour the mixture in a pan

    Place in the over for 30 minutes

Object Oriented Programming: A cake baking example

- Solution to a problem is a system of interacting **objects**
- An object has attributes and behavior
- What are the objects in this example?

1. Preheat the oven to 350F
2. Get the ingredients: 2 eggs, 1cup flour, 1 cup milk
3. Mix ingredients in a bowl
4. Pour the mixture in a pan
5. Place in the oven for 30 minutes

Objects have attributes and behavior: A cake baking example

Define classes to represent objects with similar attributes & behaviors

Object	$\boxed{\text{Attributes}}$ → coded as data members of a class	$\boxed{\text{Behaviors}}$ → coded as member functions of a class
Oven	Size Temperature Number of racks	Turn on Turn off Set temperature ✓
Bowl	Capacity Current amount	Pour into Pout out
Egg	Size	Crack Separate(white from yolk)

A class: pattern for describing similar objects

A generic pattern that is used to describe objects that have similar attributes and behaviors

e.g. a bowl and a pan may be described by the same class

```
class Dish{
    void pourIn( double amount);
    void pourOut(double amount);
    double capacity;
    double currentAmount;
};
```

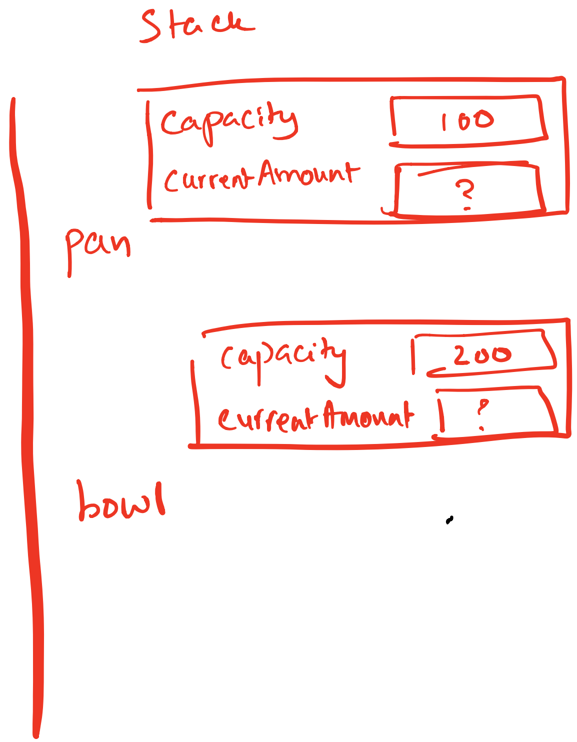
Objects vs classes

```
class Dish{
        void pourIn( double amount);
        void pourOut(double amount);
        double capacity;
        double currentAmount;
};
 //Creating objects of this class
```

```
Dish   pan;
Dish   bowl;

bowl.capacity = 200;
pan.capacity = 100;
```

instances/objects of class Dish. Each of these has its own copy of the me variables



Concept: Classes describe objects

- Every object belongs to (is an instance of) a class
- An object may have fields, or variables
  - The class describes those fields
- An object may have methods
  - The class describes those methods
- A class is like a template, or cookie cutter

Concept: Classes are like Abstract Data Types

- An Abstract Data Type (ADT) bundles together:
  - some data, representing an object or "thing"
  - the operations on that data
- The operations defined by the ADT are the *only* operations permitted on its data
- ADT = classes + information hiding

```
class Dish{
public:
     void pourIn( double amount);
     void pourOut(double amount);
private:
     double capacity;
     double currentAmount;
};
```

Approximate Terminology

- instance = object
- field = instance variable
- method = function
- sending a message to an object = calling a function

Some advice on designing classes

- Always, *always* strive for a narrow interface
- Follow the principle of information hiding:
  - the caller should know as little as possible about how the method does its job
  - the method should know little or nothing about where or why it is being called
- Make as much as possible private
- Your class is responsible for it’s own data; don’t allow other classes to easily modify it!

What we have spoken about so far?

- Class = Data + Member Functions.
- Abstract Data Type = Class + information hiding
- How to activate member functions.
- But you still need to learn how to write the bodies of a class’s methods.

Next time

- Implementing C++ classes
  - information hiding with access specifiers
  - Constructors