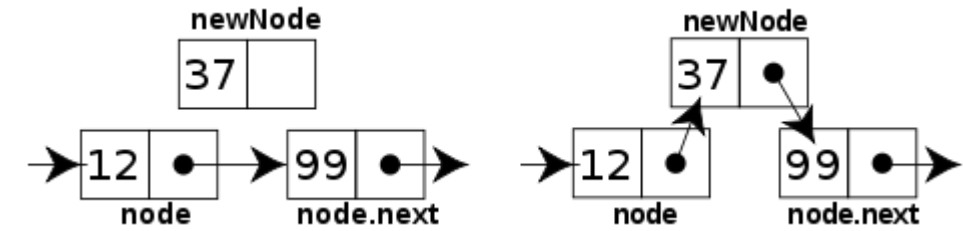


```

INSERTION-SORT(A)
1  for j = 2 to A.length
2    key = A[j]
3    // Insert A[j] into the sorted
   sequence A[1..j-1].
4    i = j - 1
5    while i > 0 and A[i] > key
6      A[i + 1] = A[i]
7      i = i - 1
8    A[i + 1] = key

```

cost	times
$c_1$	$n$
$c_2$	$n - 1$
$c_3$	$n - 1$
$c_4$	$n - 1$
$c_5$	$\sum_{j=2}^n t_j$
$c_6$	$\sum_{j=2}^n (t_j - 1)$
$c_7$	$\sum_{j=2}^n (t_j - 1)$
$c_8$	$n - 1$



# WELCOME TO CS 24!

## Problem Solving with Computers-II

Instructor: Diba Mirza

C++

```

#include <iostream>
using namespace std;

int main() {
    cout << "Hola Facebook!\n";
    return 0;
}

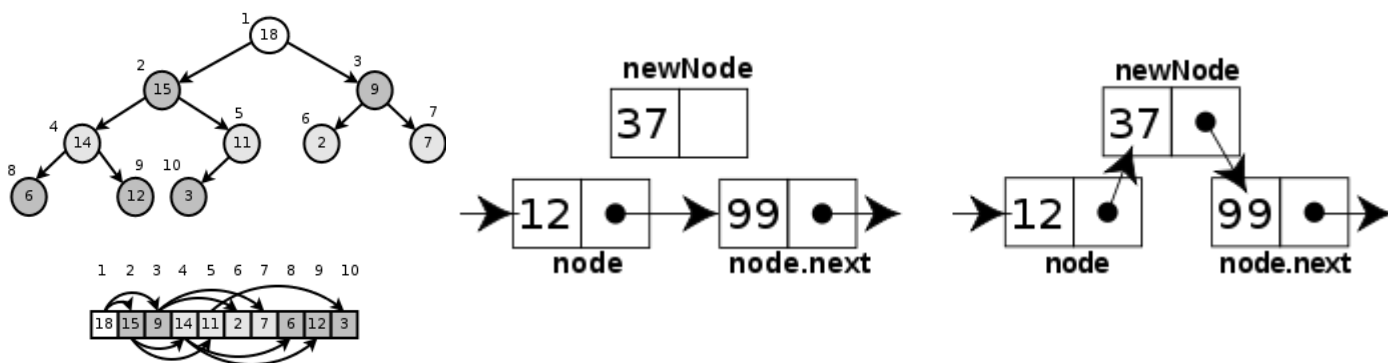
```

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

# 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
- Understand what goes on **under the hood** of programs



```

INSERTION-SORT(A)
1  for j = 2 to A.length
2    key = A[j]
3    // Insert A[j] into the sorted
   sequence A[1..j - 1].
4    i = j - 1
5    while i > 0 and A[i] > key
6      A[i + 1] = A[i]
7      i = i - 1
8    A[i + 1] = key
  
```

cost	times
$c_1$	$n$
$c_2$	$n - 1$
0	$n - 1$
$c_4$	$n - 1$
$c_5$	$\sum_{j=2}^n t_j$
$c_6$	$\sum_{j=2}^n (t_j - 1)$
$c_7$	$\sum_{j=2}^n (t_j - 1)$
$c_8$	$n - 1$

## Data Structures

## C++

## Complexity Analysis

# Course Logistics

- Course website: <https://ucsb-cs24.github.io/s21/>
- NO MAKEUPS ON QUIZZES and FINAL EXAM!
- Keep track of assignment due dates by reviewing the “weekly pattern” posted on Gauchospace
- No extensions on lab/programming assignments. Please plan to submit before the due date
- 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](mailto:help@engineering.ucsb.edu)

# About you...

What is your familiarity/confidence with Object Oriented Programming?

- 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++?

- 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.

# Feeling connected - online!

- Use the Zoom chat window to ask questions anytime
- We'll also use the chat window for discussions.
- Practice with the chat window:
  - What do you hope to get out of this class?

## 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 its 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

- C++ Memory Model, Pointers and References