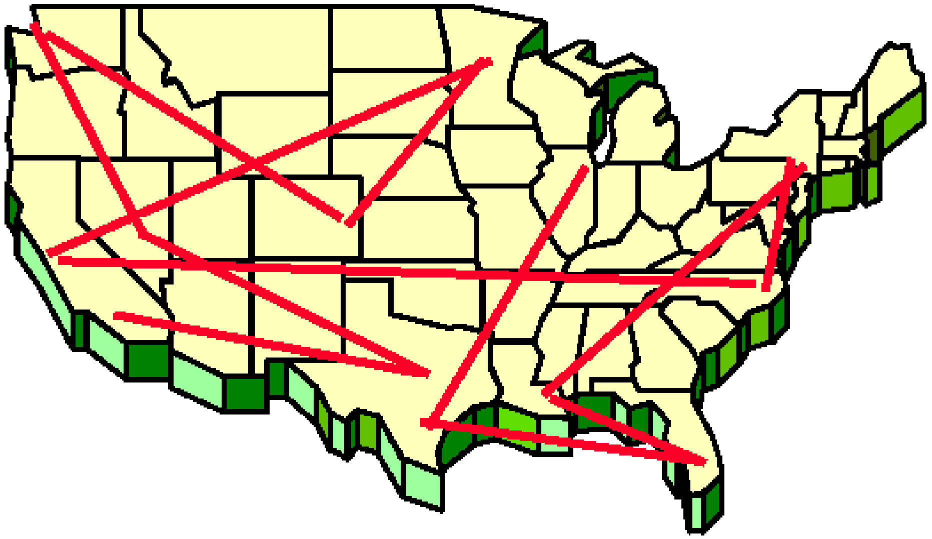
# Compsci 101
# More Recursion, Last Lecture

Susan Rodger

April 19, 2022



## **Y** is for …

- **YAML and YACC**
  - Yet Another ...
- **Y2K:** https://www.youtube.com/watch?v=rblt2EtFfC4
  - How many bits are enough bits?
- **YouTube**
  - Connected to computing ...

## `z` is for …

- **Zero**
  - There are two, or 10 bits in the universe
- **Zip**
  - Compressed file archive format
- **Zork**
  - Text-based adventure game
  - https://www.youtube.com/watch?v=TNN4VPlRBJ8

# Announcements

- Assign 6 Recommender, due today!
  - Today is last day for Office Hours
  - No consulting hours tomorrow
  - One grace day, NO LATE DAYS!
  - MUST TURN in BY tomorrow, 4/20
- Assign 7 Create due, tomorrow, April 20!
  - Grace period is through Sunday, Apr 24
  - No turnin's after that.

- Exam 4 …. Coming back later this week

## Assignment 7: More samples from previous semesters

## PFTD

- Final Exam
- Review Recursion
  - Solving a problem by solving smaller problems
- Finishing up
  - What more is there in CS?

# Final Exam – Wed, Apr 27 9am-Noon

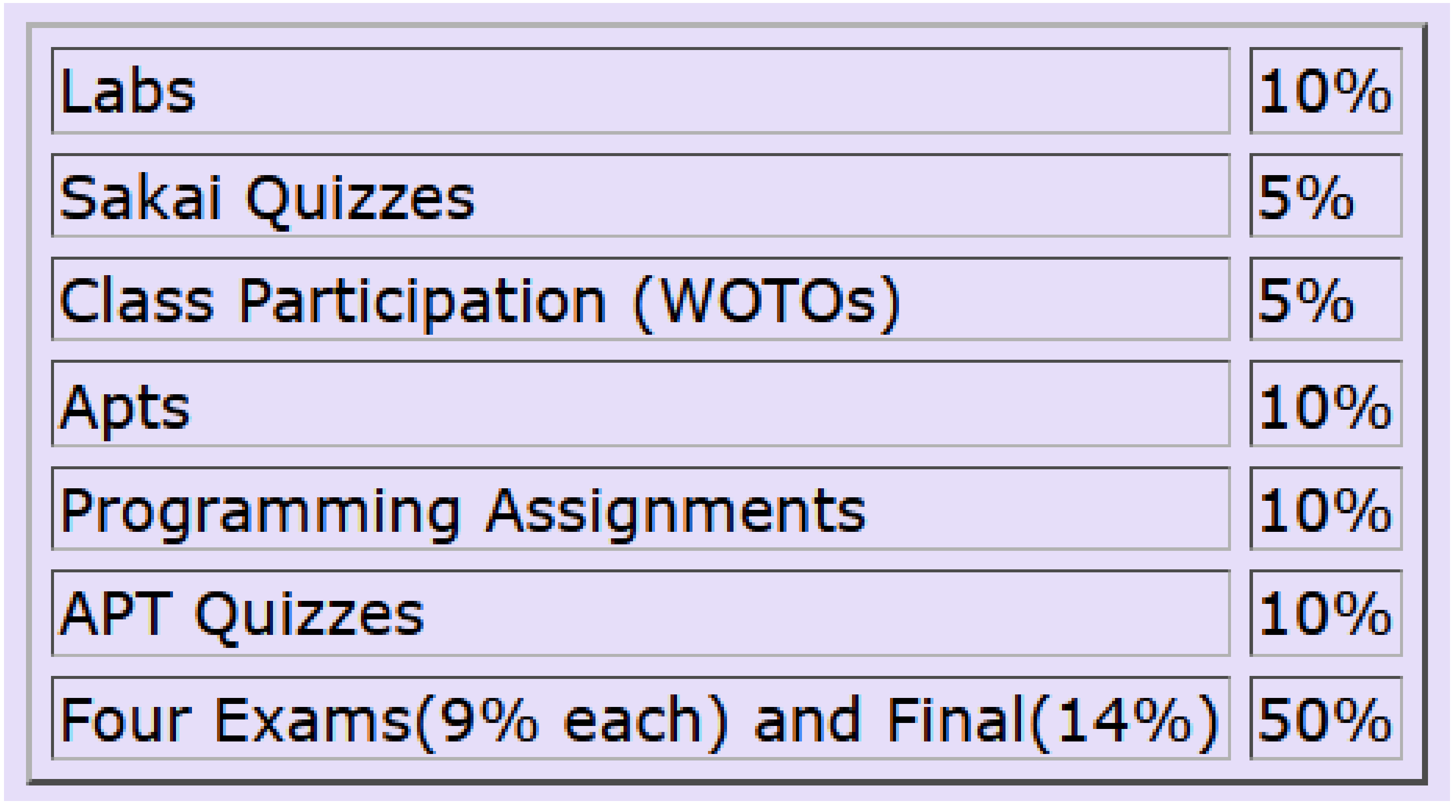
- Get a good amount of sleep!
- Set 3 alarms to wakeup!

## Final Exam

- **Study like you studied for Exam 4**
  - Use Exam 4 handout
- **We only have a little material since then**
  - Recommender
    - – this is all about stuff we did before
  - Modules
    - Exceptions
  - Recursion – reading only, no writing
- **Not on the exam**
  - Images, turtles, exceptions

# Calculate Your Grade

- From "About" tab on course web page



| | |
|---|---|
| Labs | 10% |
| Sakai Quizzes | 5% |
| Class Participation (WOTOs) | 5% |
| Apts | 10% |
| Programming Assignments | 10% |
| APT Quizzes | 10% |
| Four Exams(9% each) and Final(14%) | 50% |

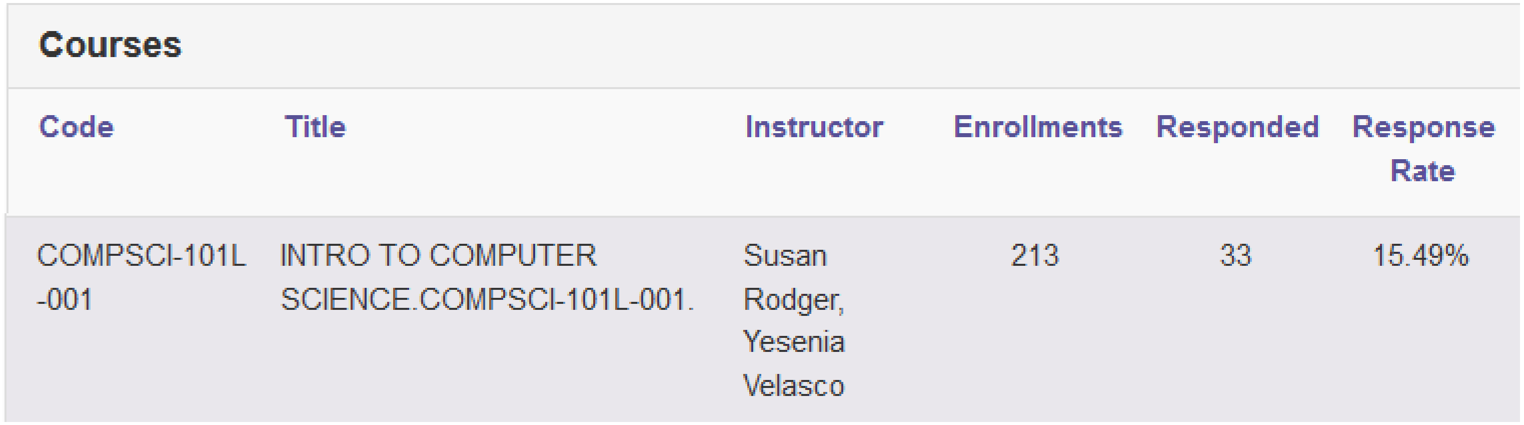
# More on Grades

- Class Participation-WOTOs – **ignore** the first two weeks (drop/add period), plus drop 10 **points**
- Sakai Quizzes **293** points– will drop **40** points
  - Your points/253, can't get more than 100
- Lab – drop **15** points (each lab is 5 pts)
- That is all we drop

# Time for CompSci 101 Course Eval

1. Please fill out Duke Course Eval on DukeHub now

Only 15% have filled it in as of last night



**Courses**

| Code | Title | Instructor | Enrollments | Responded | Response Rate |
|---|---|---|---|---|---|
| COMPSCI-101L -001 | INTRO TO COMPUTER SCIENCE.COMPSCI-101L-001. | Susan Rodger, Yesenia Velasco | 213 | 33 | 15.49% |

1. If 60% fill it out, everyone 1 extra point on final exam
2. If 75%, everyone gets 1 additional extra credit point on the Final exam

# Review: Recursion Summary

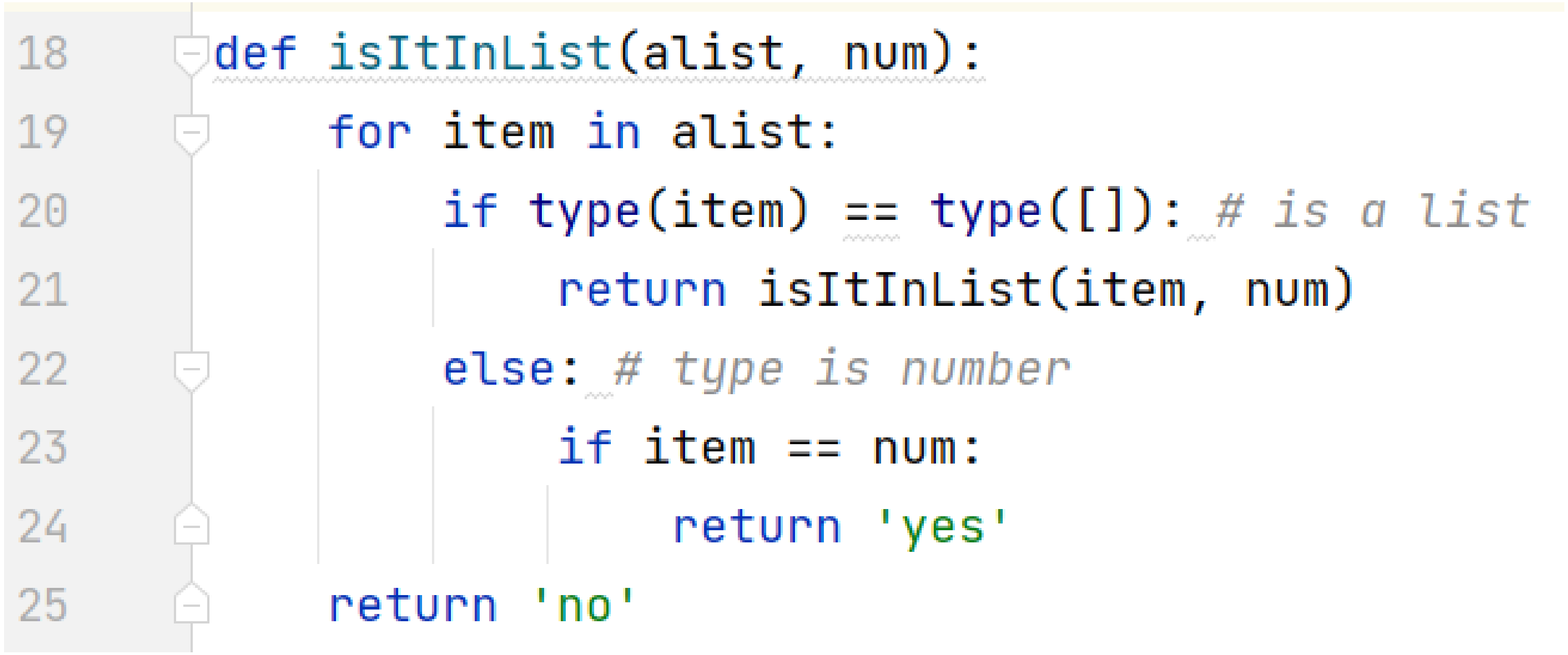
- **Make Simpler or smaller calls**
  - Call a clone of itself with different input
- **Must have a base case when no recursive call can be made**
    - Example - The last folder in the folder hierarchy will not have any subfolders. It can only have files. That forms the base case
    - This is the way out of recursion!

# Problem: is a number in a list?

- Is 5 in [7, 5, 6, 8] ?

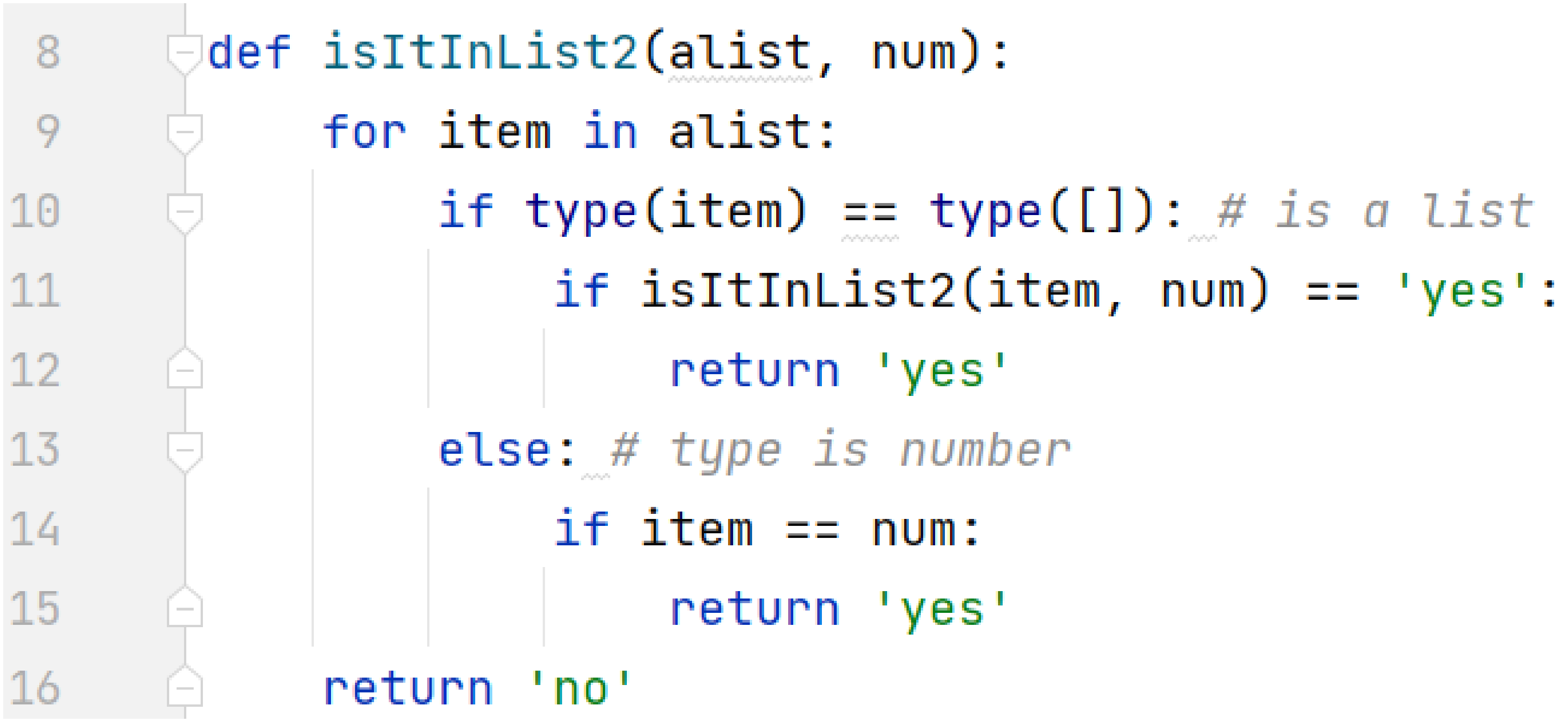
- Is 8 in [5, [ [7,4], 9, [3, 4]], [4, [5, [2, [8, 1], 4, ] ], 5] ] ?

# Possible solution



```
def isItInList(alist, num):
    for item in alist:
        if type(item) == type([]): # is a list
            return isItInList(item, num)
        else: # type is number
            if item == num:
                return 'yes'
    return 'no'
```

## Possible Solution 2



```python
def isItInList2(alist, num):
    for item in alist:
        if type(item) == type([]): # is a list
            if isItInList2(item, num) == 'yes':
                return 'yes'
        else: # type is number
            if item == num:
                return 'yes'
    return 'no'
```

## Problem: is a number in a list?

- Is 5 in [7, 5, 6, 8] ?
- Is 8 in [5, [ [7,4], 9, [3, 4]], [4, [5, [2, [8, 1], 4, ] ], 5] ] ?

# Revisit the APT Bagels Recursively

```
filename:  Bagels.py

def bagelCount(orders) :
    """
    return number of bagels needed to fulfill
    the orders in integer list parameter orders
    """
```

1. 
```
orders = [1,3,5,7]

Returns:  16
```

No order is for more than a dozen, return the total of all orders.

2. 
```
orders = [11,22,33,44,55]

Returns: 175 since 11 + (22+1) +(33+2) + (44+3) + (55+4) = 175
```

## APT Bagels Recursively
## bit.ly/101s22-0419-1

## A peek into CompSci 201

- Sorting  list of numbers
- Quicksort algorithm
    - Uses recursion

# Quicksort - Idea

- Pivot – select and adjust the list
  - Select one of the elements
  - Put it where it belongs in sorted order
  - Put elements less than it, to its left
  - Put elements greater than it, to its right
- Recursively sort the elements to its left
- Recursively sort the elements to its right
- Done!

# Quicksort - Idea

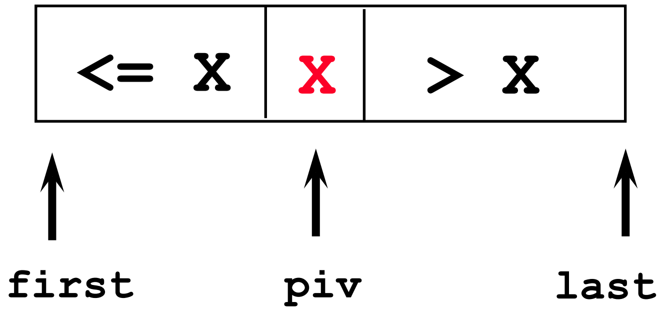
- Pivot – select and adjust list
- Recursively sort the elements to its left
- Recursively sort the elements to its right

$\bigcirc\!\!\!\!\!\!5$ 9 1 4 3 6 2 7

# Quicksort: fast in practice

```
def doQuick(list, first, last) {
    if (first >= last) return

    piv = pivot(list,first,last)
    doQuick(list,first,piv-1)
    doQuick(list,piv+1,last)
}
```

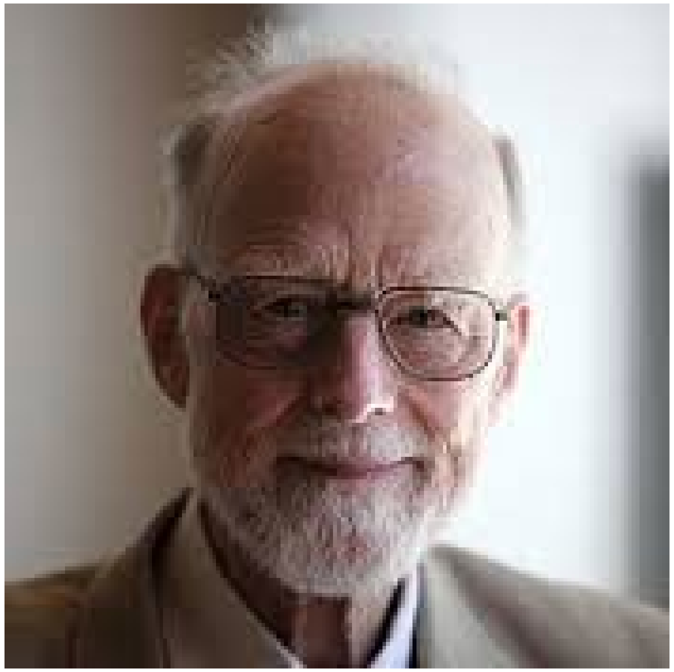


# Sir Anthony (Tony) Hoare

Invented Quicksort in 1962
- he didn't get recursion

Turing Award winner
- programming language design
- Algol 60

"There are two ways of constructing a software design. One way is to make it so simple that there are obviously no deficiencies. And the other way is to make it so complicated that there are no obvious deficiencies."



"Inside every large program is a small program struggling to get out."

# What is Computing? Informatics?

- **What is computer science, what is its potential?**
  - What can we do with computers in our lives?
  - What can we do with computing for society?
  - Will networks transform thinking/knowing/doing?
  - Society affecting and affected by computing?
  - Changes in science: biology, physics, chemistry, …
  - Changes in humanity: access, revolution (?), …

- **Privileges and opportunities available if you know code**
  - Writing and reading code, understanding algorithms
  - Majestic, magical, mathematical, mysterious, …

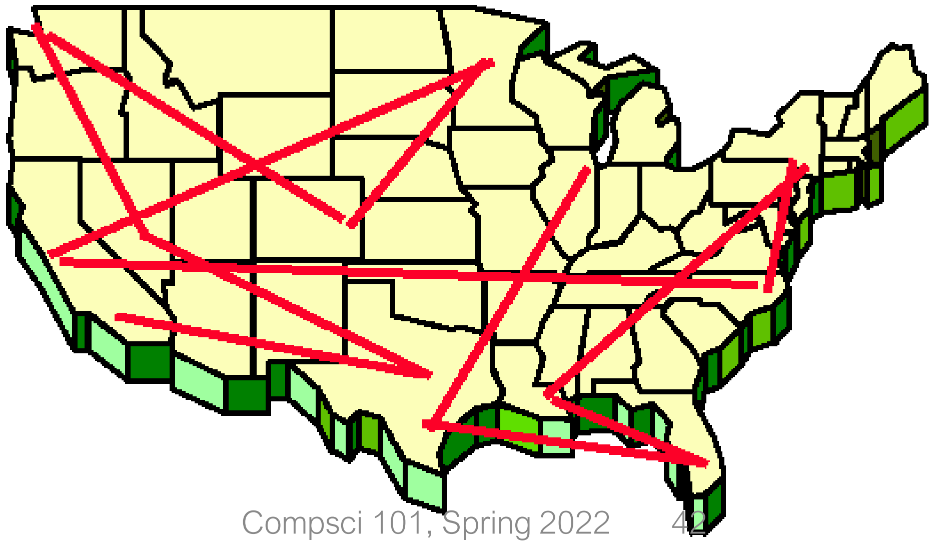
# Computing - solve all problems?

- **Some problems can be solved 'efficiently'**
  - Run large versions fast on modern computers
  - What is 'efficient'? It depends
- **Some cannot be solved by computer.**
  - Provable! We can't wait for smarter algorithms

- **Some problems have no efficient solution**
  - Provably exponential $2^n$ so for "small" n …
- **Some have no known efficient solution, but**
  - If one does they all do!

## Problem: Traveling Band

- Band wants you to schedule their concerts.
- They don't like to travel. Minimize the time they are on the bus!
- Given N cities, what is the best schedule (shortest distance) to visit all N cities once?



## How do you calculate the best path?

- Try all paths
  - Atlanta, Raleigh, Dallas, Reno, Chicago
    - Add up the distance in this order
  - Dallas, Atlanta, Raleigh, Reno, Chicago
    - Add up the distance in this order
  - Etc.
- Would you agree to code this up?

# Traveling Band questions
# bit.ly/101s22-0419-2

# How long?

| Number of Cities | All paths – N! | Time to solve - $10^9$ Instructions per second |
|---|---|---|
| 10 | 3 million | |
| 15 | $10^{12}$ | |
| 18 | $10^{15}$ | |
| 20 | $10^{18}$ | |
| 25 | $10^{25}$ | |

## How is Python like all other programming languages, how is it different?

## Find all unique/different words in a file, in sorted order

# Unique Words in Python

```
def main():
    f = open('/data/melville.txt', 'r')
    words = f.read().strip().split()
    allWords = set(words)

    for word in sorted(allWords):
        print(word)

if __name__ == "__main__":
    main()
```

## Unique words in Java

```
import java.util.*;
import java.io.*;
public class Unique {
  public static void main(String[] args)
                              throws IOException{
    Scanner scan =
            new Scanner(new
   File("/data/melville.txt"));
    TreeSet<String> set = new TreeSet<String>();
    while (scan.hasNext()){
        String str = scan.next();
        set.add(str);
    }
    for(String s : set){
        System.out.println(s);
    }
  }
}
```

## Unique words in C++

```
#include <iostream>
#include <fstream>
#include <set>
using namespace std;

int main(){
  ifstream input("/data/melville.txt");
  set<string> unique;
  string word;
  while (input >> word){
    unique.insert(word);
  }
  set<string>::iterator it = unique.begin();
  for(; it != unique.end(); it++){
    cout << *it << endl;
  }
  return 0;
}
```

## Unique words in PHP

```
<?php

$wholething = file_get_contents("file:///data/melville.txt");
$wholething = trim($wholething);

$array = preg_split("/\s+/",$wholething);
$uni = array_unique($array);
sort($uni);
foreach ($uni as $word){
   echo $word."<br>";
}

?>
```

## What is next?

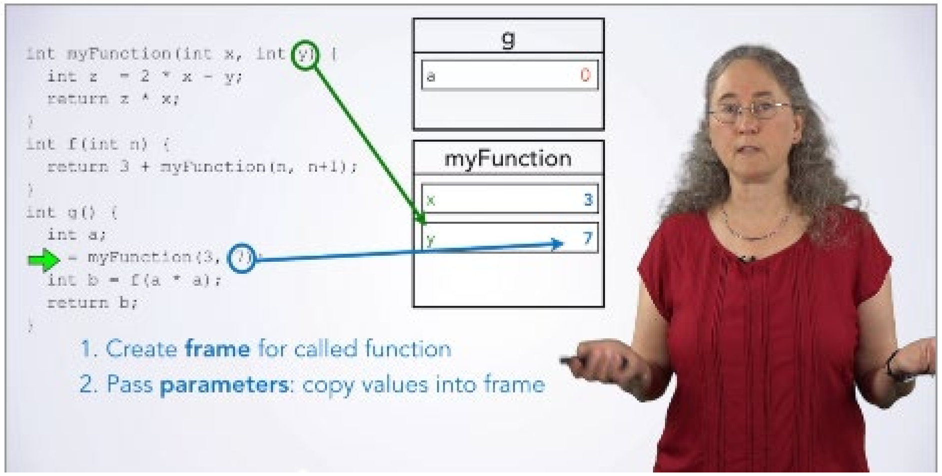
- CompSci 201
  - Java, efficiency, other ways to organize data
- CompSci 230 – can take concurrently with 201
  - Discrete Mathematics
    - Course substitutions if you take a lot of math/stats
- CompSci 260 Computational Biology
- CompSci 216 Everything Data
- CompSci 240 Race, Gender, Class and Computing

## What to do in the summer?
## Take a Duke Coursera course Free

- Course on Java

VIDEO

### Functions

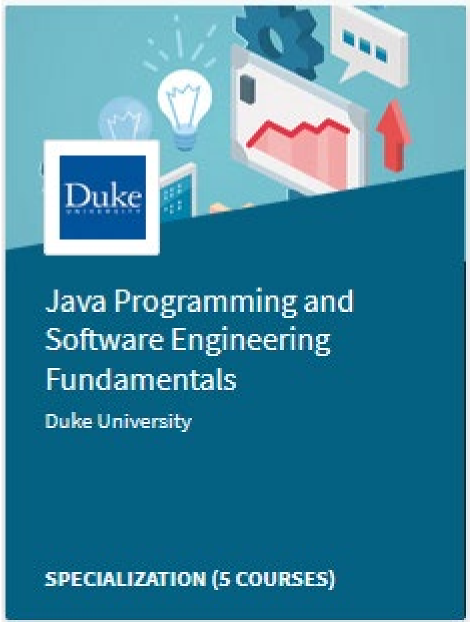


```
int myFunction(int x, int y) {
  int z  = 2 * x - y;
  return z * x;
}
int f(int n) {
  return 3 + myFunction(n, n+1);
}
int g() {
  int a;
  a = myFunction(3, 7);
  int b = f(a * a);
  return b;
}
```

1. Create **frame** for called function
2. Pass **parameters**: copy values into frame

# Duke Coursera course on Java

- Coursera for Duke
  - https://online.duke.edu/coursera-for-duke/
- Java Programming/Soft Eng Fund
  - 5 courses
    - HTML/CSS/JavaScript
    - 4 courses on Java
- Course is FREE for Duke students
  - Go through link above


Duke
Java Programming and Software Engineering Fundamentals
Duke University
SPECIALIZATION (5 COURSES)

## End with A CS Story
## bit.ly/101s22-0419-3