



TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU



TEXAS

The University of Texas at Austin

# Hack@TACC - REU Coding Challenge

PRESENTED BY:

# Introductions - Icebreaker



Charlie Dey (*TACC*)



Je'aime Powell (*TACC*)

# Computational Thinking Objectives

The student will ...

- Learn the about the concept of “computational thinking”
- Practice algorithm implementation through abstraction
- Learn about the concept of pseudo code
- Apply computational thinking to the equation for a straight line



# Back when mathematicians were computers and computers were calculators...

- Initially all programming was dedicated to translating math formulas.
- The work lead to the language FORMula TRANslation.



# “Computational Thinking is the translation of ideas into computer code” ~Victor Eijkhout

## Mathematical Thinking

- Number of people an elevator takes per day
- Speed (velocity) of an elevator
- Distribution of people in an elevator

## Computational Thinking

- If there are  $X$  # of people expected to use elevators, how many should be installed?
- If someone at floor 0 presses the call button and there are available cars on floors 5 and 9, which car should respond?



# The Process of Forming Logic (Think teaching a 3 year-old)

How would you tell a three(3) year old family member to get your keys out of the drawer in your room ?



# Requirements, Logic, Algorithms, and Parameters

**Requirements** - what elements are needed before the job can be taken on

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

7

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

# Making a PB&J Sandwich

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]





# Finding a definition in a dictionary

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

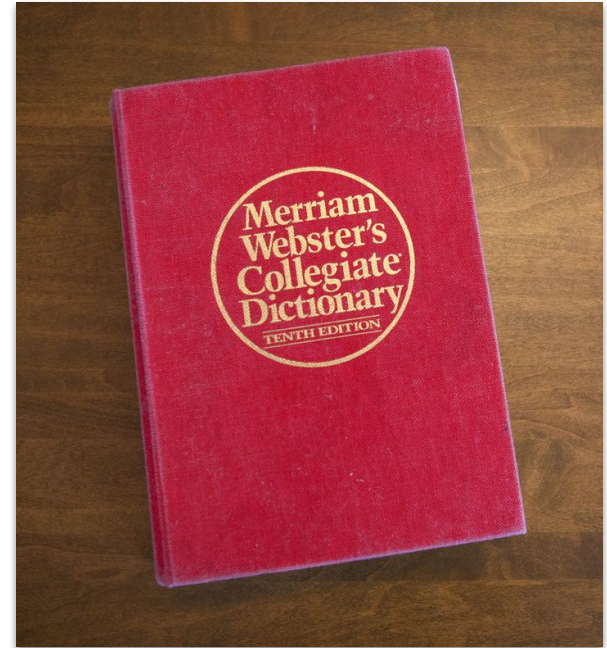
## Given (Input):

- Dictionary
- Word (string)

## Find (Output):

- Definition

*Define your algorithm*



# Sorting

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

**Given:**

A bag of potatoes



**Problem:**

Sort the bag of potatoes from smallest to largest

**Algorithm:????**

10



# What would we need to solve for “y”

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

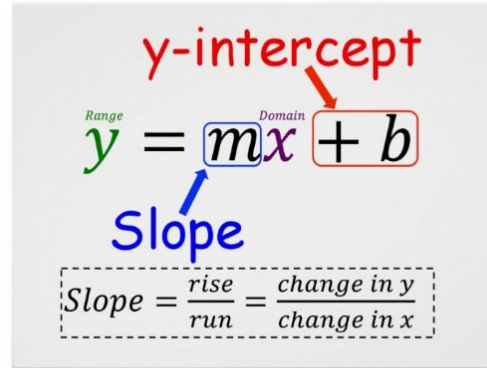
Problem (function):

$$y=mx+b$$

Define the **Input**  
(parameters)

Define the **Output**  
(parameters)

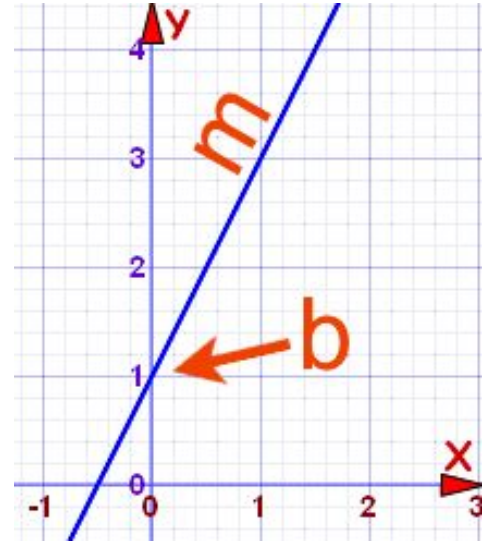
Define the **Algorithm**



The diagram shows the equation  $y = mx + b$  with several annotations. The variable  $y$  is green and labeled "Range". The variable  $x$  is purple and labeled "Domain". The coefficient  $m$  is blue and labeled "Slope". The constant  $b$  is red and labeled "y-intercept". A red arrow points from the text "y-intercept" to the  $b$  term. A blue arrow points from the text "Slope" to the  $m$  term. The equation is enclosed in a dashed box.

$$y = mx + b$$

*Slope* =  $\frac{\text{rise}}{\text{run}} = \frac{\text{change in } y}{\text{change in } x}$



# Think outside the "Box"

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

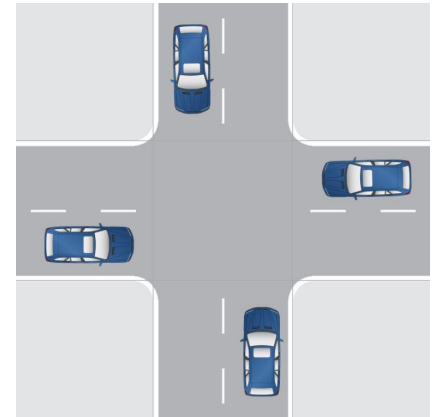
**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

**Problem:**

4 automated cars come to an intersection at the same time.

Who goes first?

**Algorithm:** ???



# What decisions did you make?

**Requirements** - what elements are needed before the job can be taken on

**Parameters** - a limit or boundary that defines the scope of a particular process or activity [limits set on an algorithm = parameters]

**Logic** - a system or set of principles underlying the arrangements of elements in a computer or electronic device so as to perform a specified task [an order in which to do a task]

**Algorithm** - a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer [logic + calculations = algorithm]

What was the last meal you ate?

What were the defining parameters on why you chose that meal?

# Where do we go from here?

Look at each problem you are going to tackle, and figure out the requirements - what is needed to solve? Figure out the logic on how to solve it, and apply the algorithm.

# Think about a Scientific Process

Let's meet Joe.

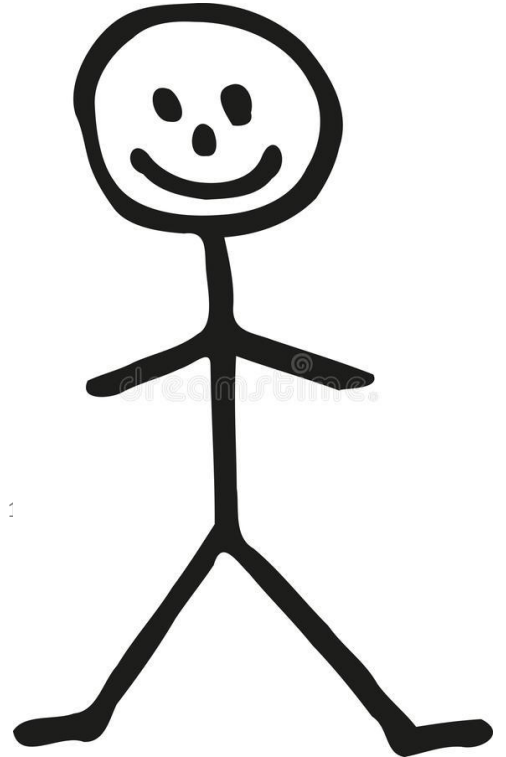
Joe might get sick.

Joe will be sick for 5 days.

After 5 days, Joe gets better.

Once Joe gets better, Joe can no longer get sick.

How would we "code" Joe?



# \*\*\*Task 1\*\*\* - Code Joe

**Variables** to hold data

**Mathematical Operations** to do math :)

**Conditionals** to make decisions

**Loops** to repeat our process

**Functions/Subroutines** to reuse code

**Objects or Classes** to define our "things"

Let's meet Joe.

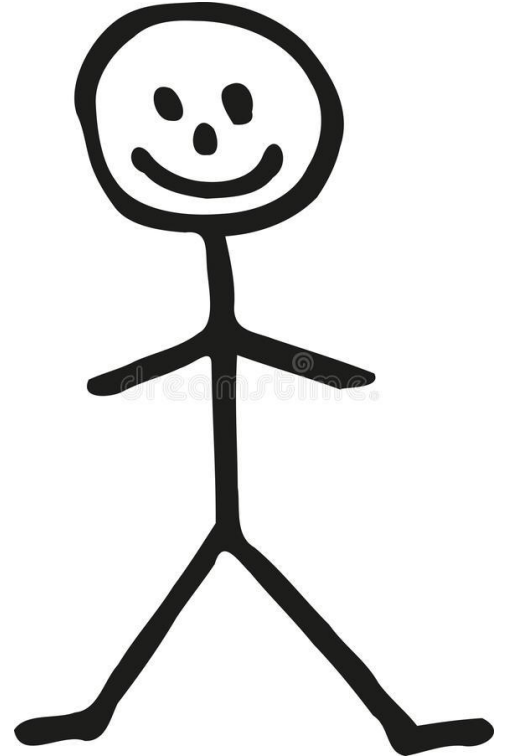
Joe might get sick.

Joe will be sick for 5 days.

After 5 days, Joe gets better.

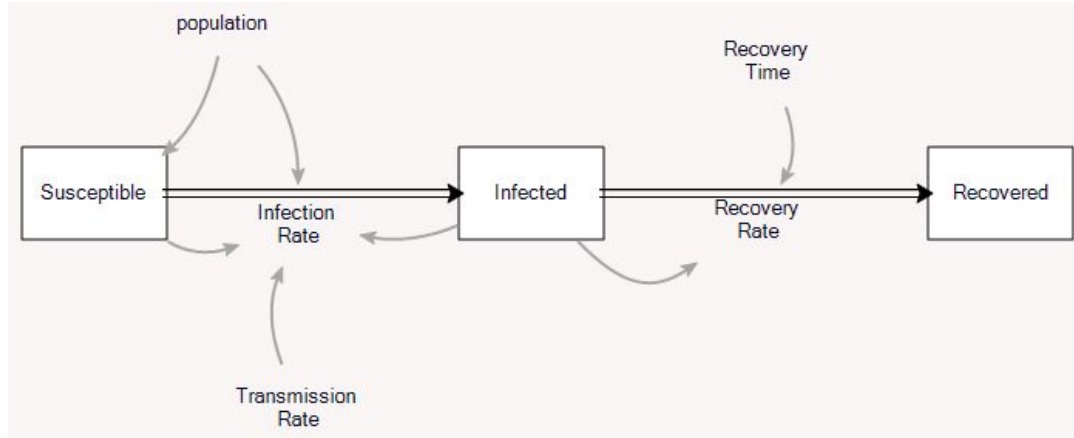
Once Joe gets better, Joe can no longer get sick.

Let's "code" Joe.





# The SIR Model



## \*\*\*Task 2\*\*\* Code Joe and Jane

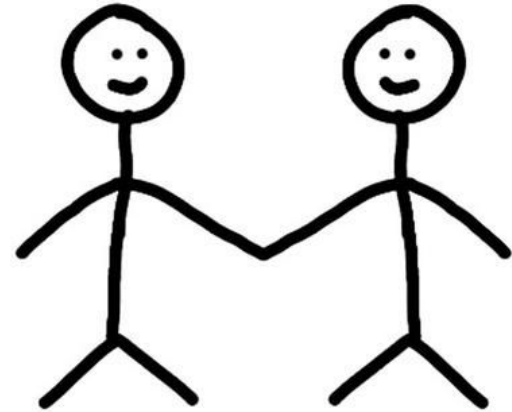
Let's meet Joe.

Joe has a friend, Jane

If Joe gets sick, Jane might get sick.

Modify your code, so when Joe gets sick that triggers Jane to roll a random number to see if she gets sick.

Loop through your code until both Joe and Jane get sick and they each get better.



# \*\*\*Task 3\*\*\* Populations and Interactions

We need more than 2 people in our "world"

People interact with other people

1. create a list of 100 Persons (Our Population)
2. code up a way so each Person interacts with 10 random people from the Population

example output:

Person 0 interacted with Persons: [5 2 100 408 768 987...]

Person 1 interacted with Persons: [67 875 356 10 299 9...]

etc...

