# CSCI 141: Computer Programming I

Lecture 1
Introduction, Logistics, Hello World

# Today

- What is this course about?
- Why are we here?
- Who is this character?
- Some course logistics
- Let’s write some code already!

# What is this course about?

- What is this course about?

- What will you learn?

# What is this course about?

**From the course catalog:**

Basic concepts of computer programming using an object oriented programming language. Topics covered: introduction to the development environment, introduction to algorithms, elements of a programming language, including data types, packages, control structures, procedures and functions, basic input and output, arrays and records, text files, strings, variant records. Algorithm development, problem solving and software engineering are emphasized.

# What is this course about?

**From the course catalog:**

Basic concepts of computer programming using an object oriented programming language. Topics covered: introduction to the development environment, introduction to algorithms, elements of a programming language, including data types, packages, control structures, procedures and functions, basic input and output, arrays and records, text files, strings, variant records. Algorithm development, **problem solving and software engineering are emphasized**.

# What is this course about?

Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

**Can you define any of these terms?**

# What is this course about?

Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

**Can you define any of these terms?**

# What is this course about?

## Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

**Can you define any of these terms?**

These are the "nuts and bolts": the *syntax* and *semantics* of programming languages.

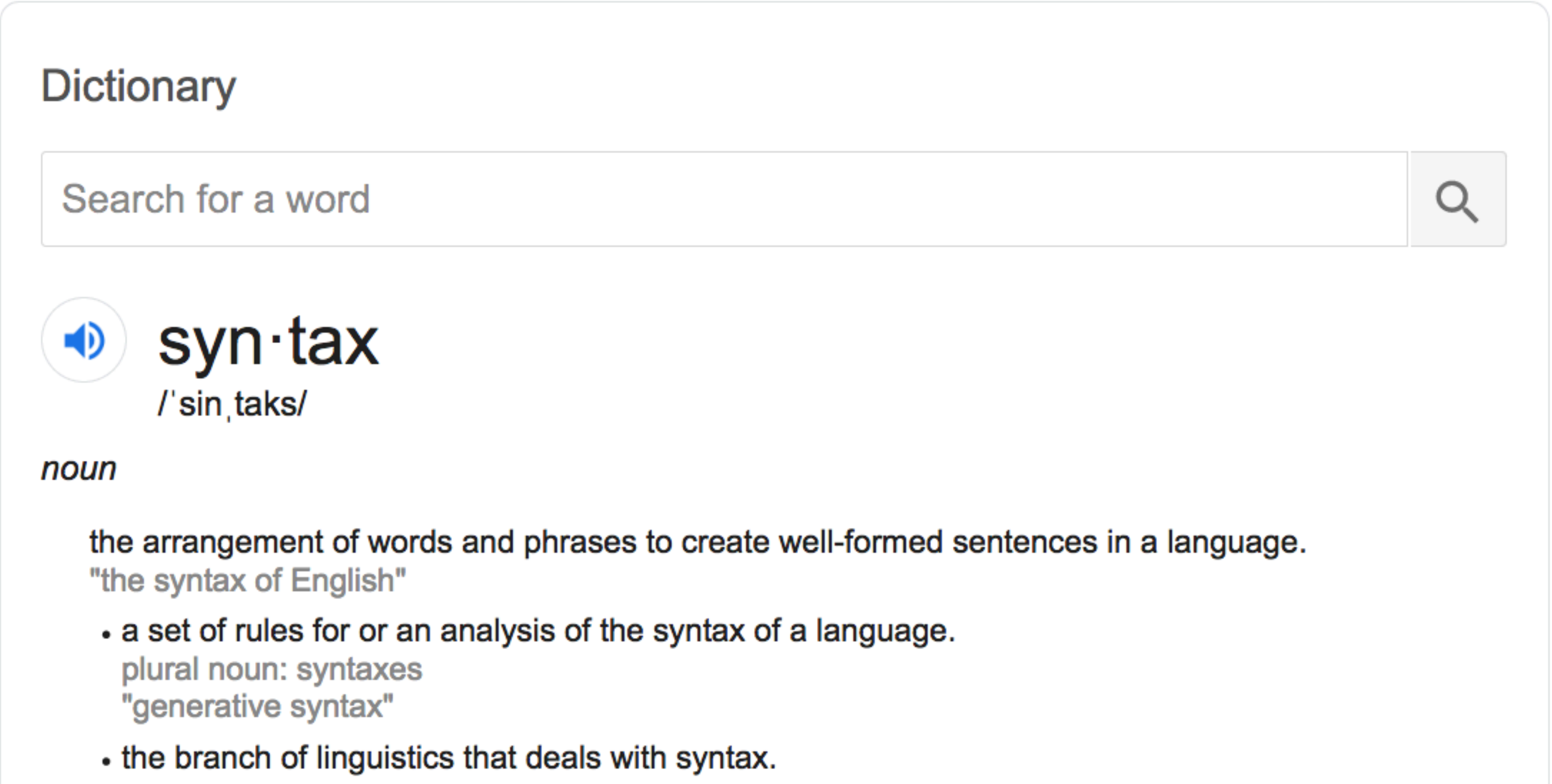
# What is this course about?

## Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

**Can you define any of these terms?**

These are the “nuts and bolts”: the *syntax* and *semantics* of programming languages.



syn·tax
/ˈsinˌtaks/

*noun*

the arrangement of words and phrases to create well-formed sentences in a language.
"the syntax of English"

- a set of rules for or an analysis of the syntax of a language.
  plural noun: syntaxes
  "generative syntax"
- the branch of linguistics that deals with syntax.

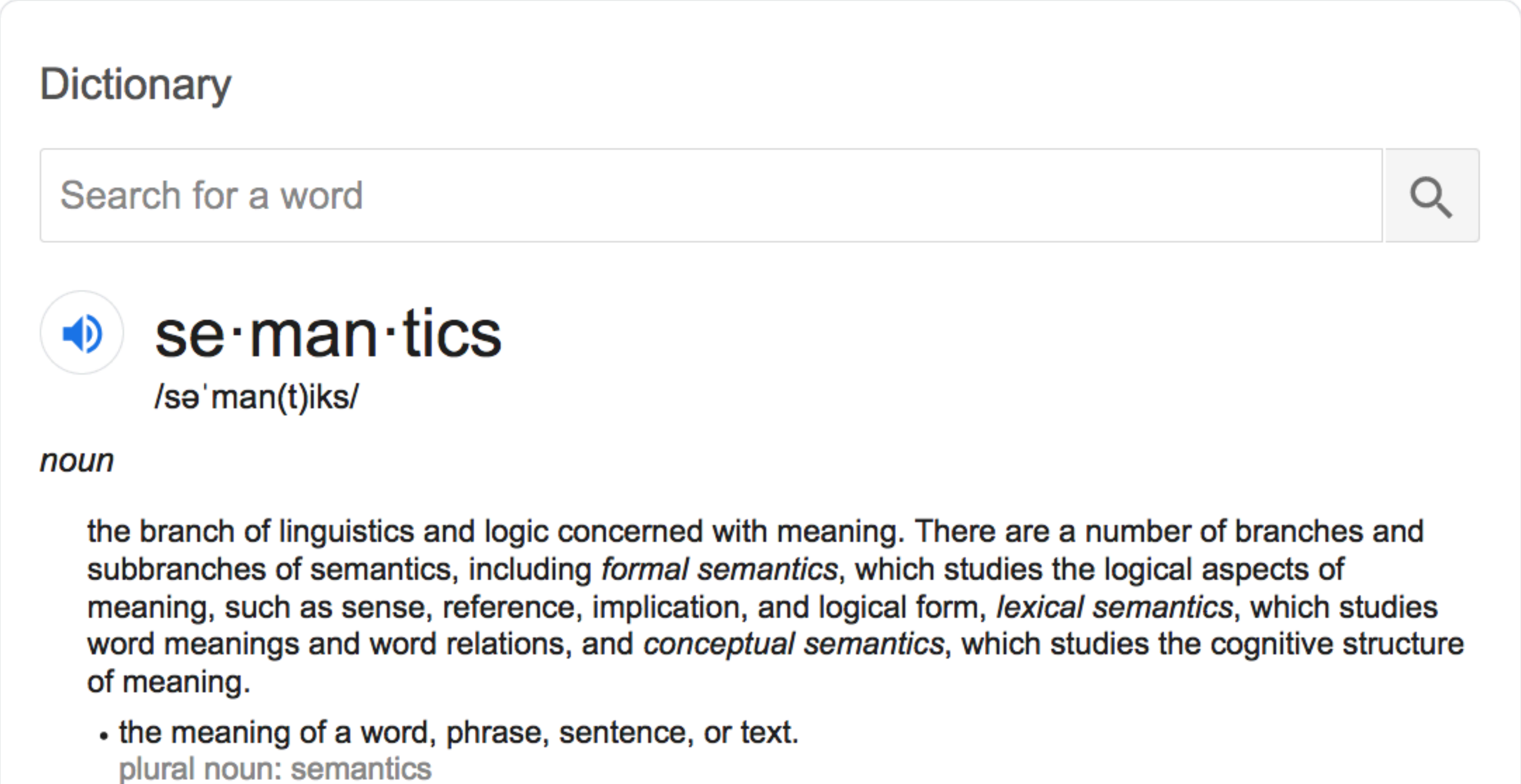
# What is this course about?

## Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

**Can you define any of these terms?**

These are the “nuts and bolts”: the *syntax* and *semantics* of programming languages.



se·man·tics

/səˈman(t)iks/

*noun*

the branch of linguistics and logic concerned with meaning. There are a number of branches and subbranches of semantics, including *formal semantics*, which studies the logical aspects of meaning, such as sense, reference, implication, and logical form, *lexical semantics*, which studies word meanings and word relations, and *conceptual semantics*, which studies the cognitive structure of meaning.

- the meaning of a word, phrase, sentence, or text.
  plural noun: semantics

# What is this course about?

**From the course catalog:**

Basic concepts of computer programming using an object oriented programming language. Topics covered: introduction to the development environment, introduction to algorithms, elements of a programming language, including data types, packages, control structures, procedures and functions, basic input and output, arrays and records, text files, strings, variant records. Algorithm development, **problem solving and software engineering are emphasized**.

# What is this course about?

Problem Solving and Software Engineering:

- Break down and analyze problems
- Design algorithms that solve problems
- Describe algorithms in pseudocode
- Implement algorithms using clearly written, correct Python code.
- Fix errors and make changes to the code once it’s written.

# What is this course about?

Problem Solving and Software Engineering:

- Break down and analyze problems
- Design algorithms that solve problems
- Describe algorithms in pseudocode
- Implement algorithms using clearly written, correct Python code.

Computer Programming:

- data types
- control structures
- functions
- strings
- arrays

- Fix errors and make changes to the code once it’s written.

# Why learn to program?

- Why do you want to learn how to program?

# Why learn to program?

# Why learn to program?

- Some ideas:

# Why learn to program?

- Some ideas:
    - Get a job with cool perks and a high salary

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary

Microkitchen at Facebook: Free snacks!

# Why learn to program?

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks

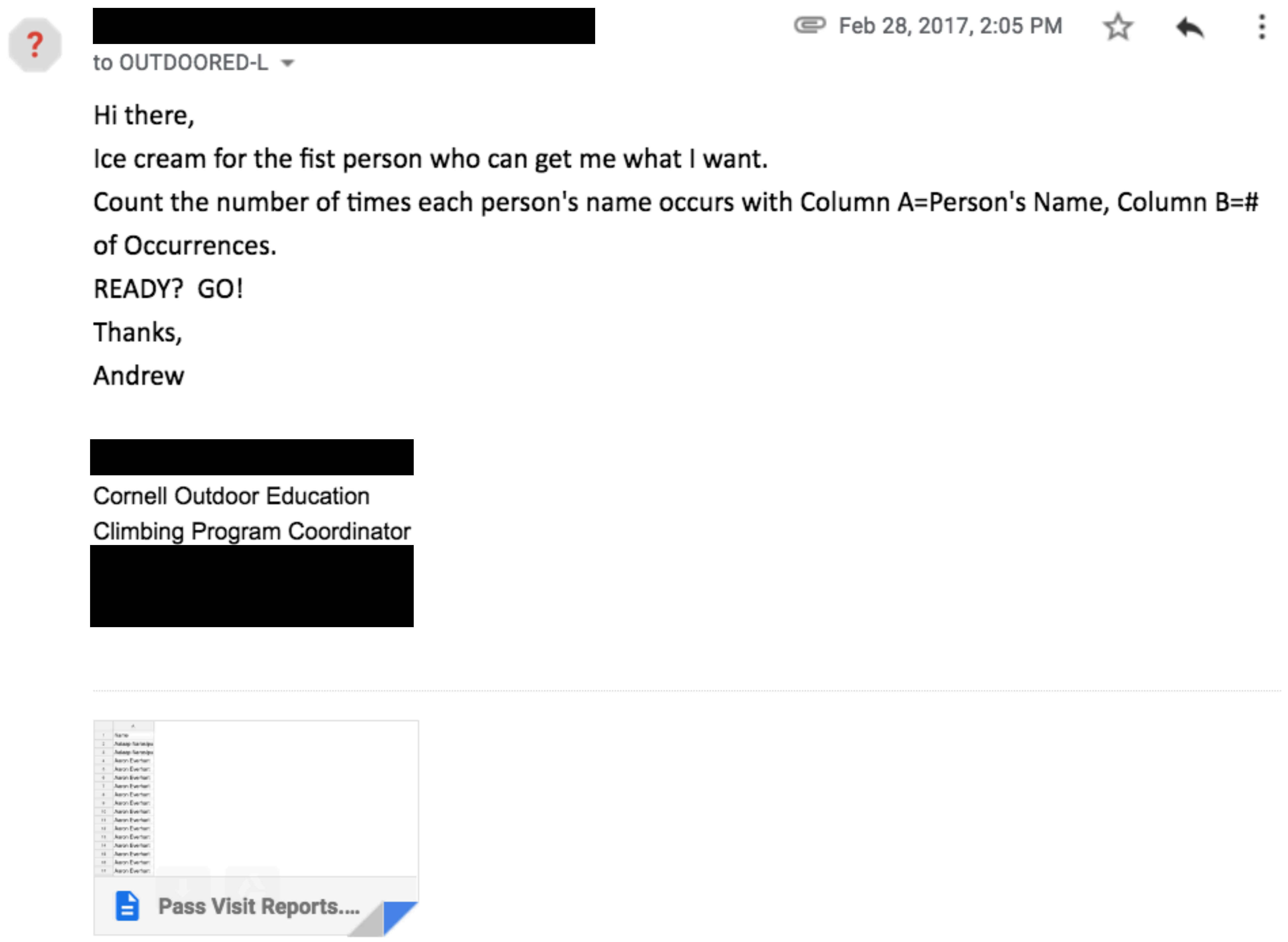
# Why learn to program?

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks
  - Process or analyze data you encounter in your chosen profession



[redacted]

to OUTDOORED-L

Feb 28, 2017, 2:05 PM

Hi there,

Ice cream for the fist person who can get me what I want.

Count the number of times each person's name occurs with Column A=Person's Name, Column B=# of Occurrences.

READY?  GO!

Thanks,

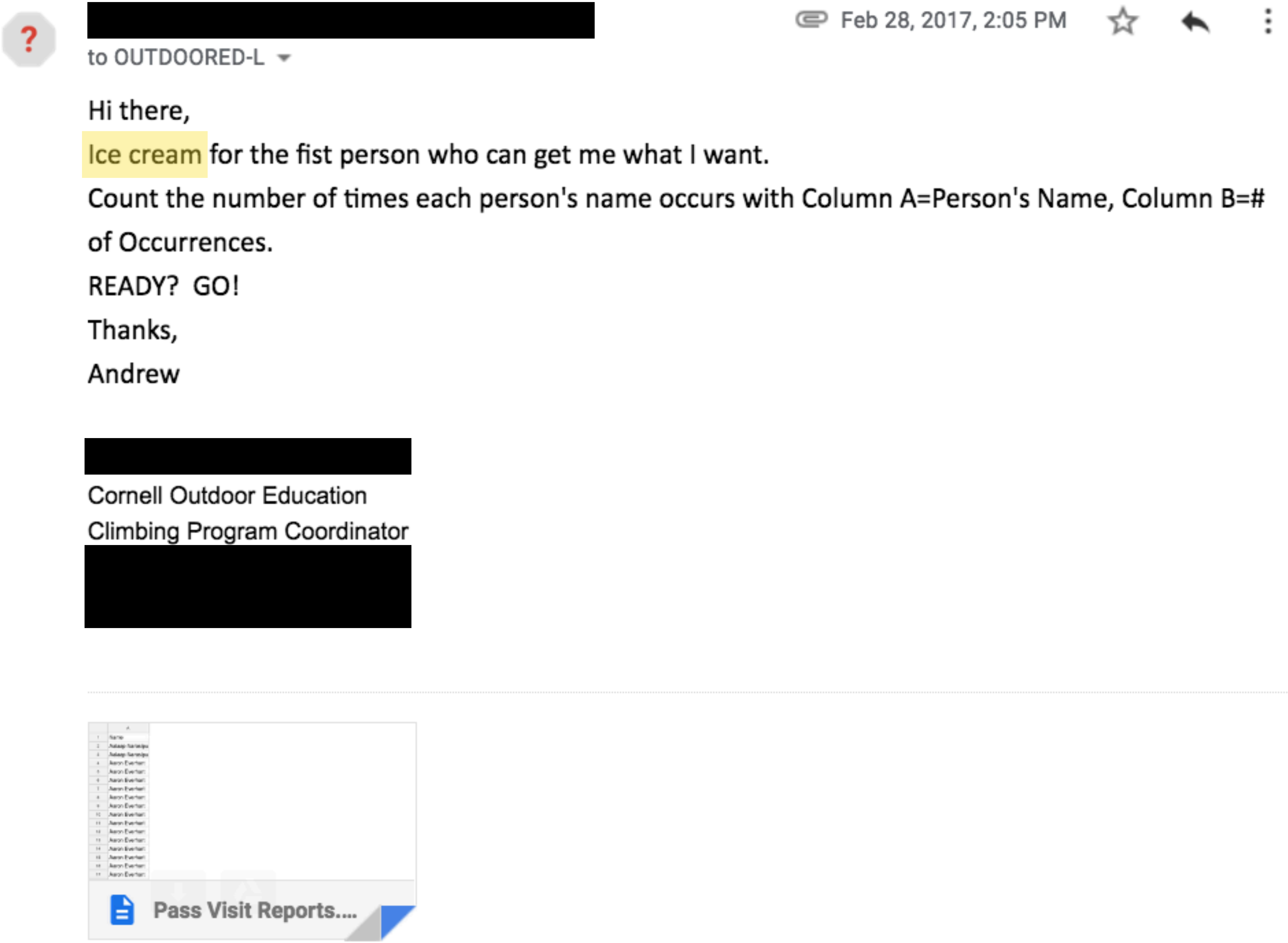
Andrew

[redacted]

Cornell Outdoor Education
Climbing Program Coordinator

[redacted]

Pass Visit Reports....



[redacted]

to OUTDOORED-L

Feb 28, 2017, 2:05 PM

Hi there,

Ice cream for the fist person who can get me what I want.

Count the number of times each person's name occurs with Column A=Person's Name, Column B=# of Occurrences.

READY? GO!

Thanks,

Andrew

[redacted]

Cornell Outdoor Education
Climbing Program Coordinator

[redacted]

Pass Visit Reports....

# Why learn to program?

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks
  - Process or analyze data you encounter in your chosen profession

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks
  - Process or analyze data you encounter in your chosen profession
  - Execute your creative vision

# Why learn to program?

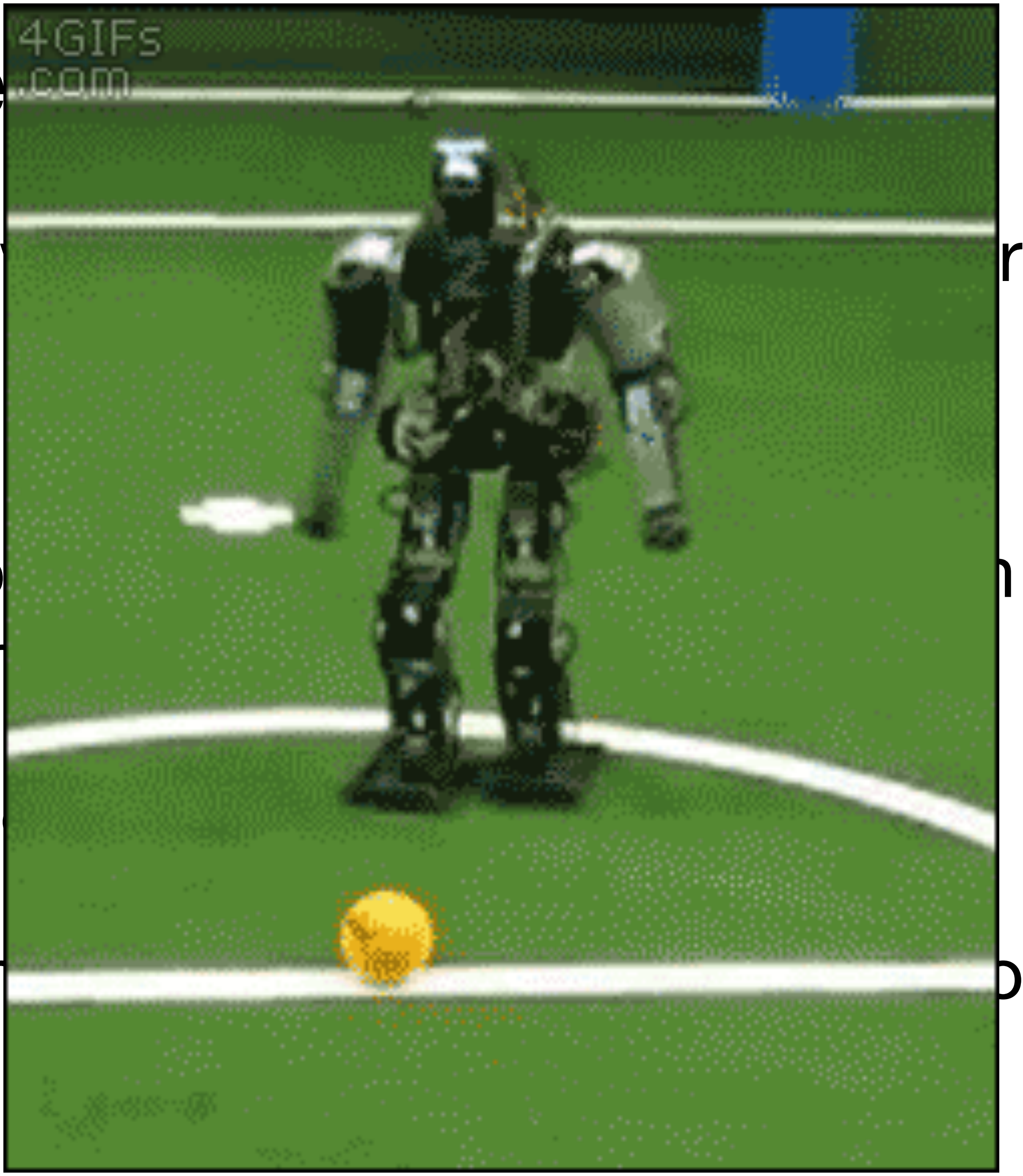
- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks
  - Process or analyze data you encounter in your chosen profession
  - Execute your creative vision
  - Understand what’s going on inside the computers you use daily

# Why learn to program?

- Some ideas:
  - Get a job with cool perks and a high salary
  - Automate repetitive tasks
  - Process or analyze data you encounter in your chosen profession
  - Execute your creative vision
  - Understand what’s going on inside the computers you use daily
  - Make friends with our future robot overlords

# Why learn to program?

- Some ide
  - Get a job ry
  - Automate
  - Process o n your chosen profession
  - Execute y
  - Understan omputers you use daily
  - Make friends with our future robot overlords



# Who is this character?

# About Me

## Scott

# About Me

## Scott Wehr

# About Me

## Scott Wehrwein

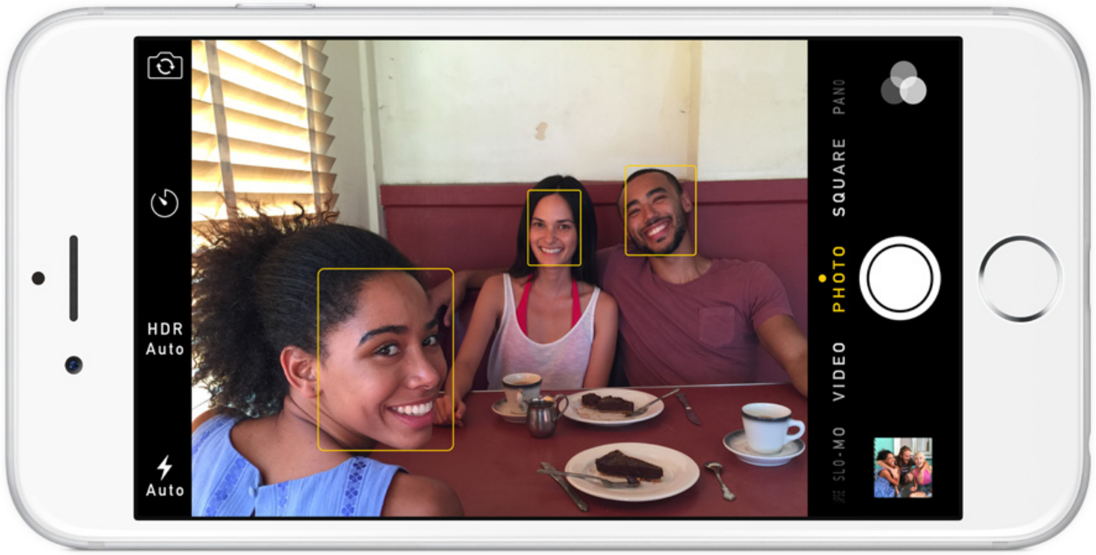
📍 🍷

# About Me

## Scott Wehrwein

📍 🍷

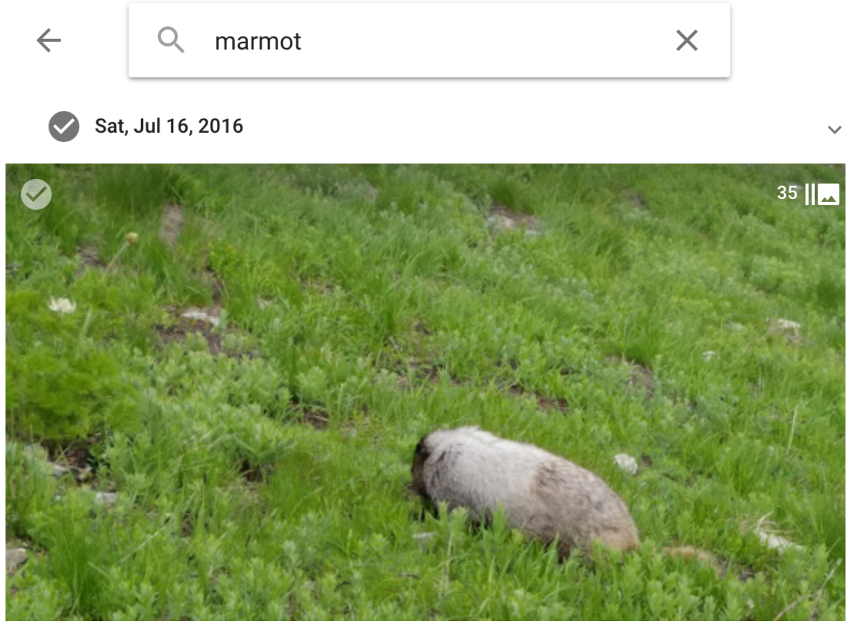
# Computer Vision: Familiar Examples



In-Camera Face Detection

Autonomous Driving

Panorama Stitching

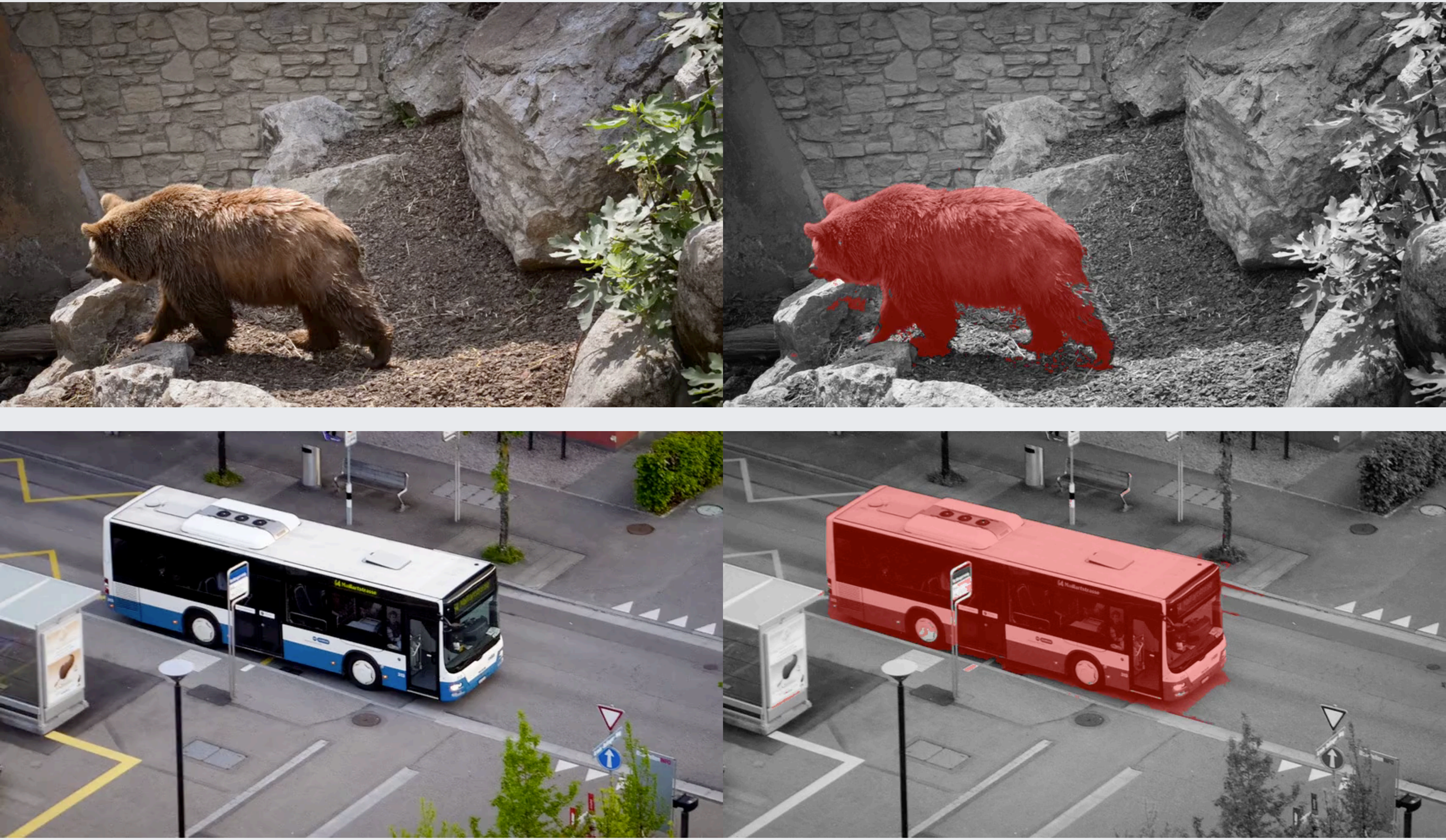


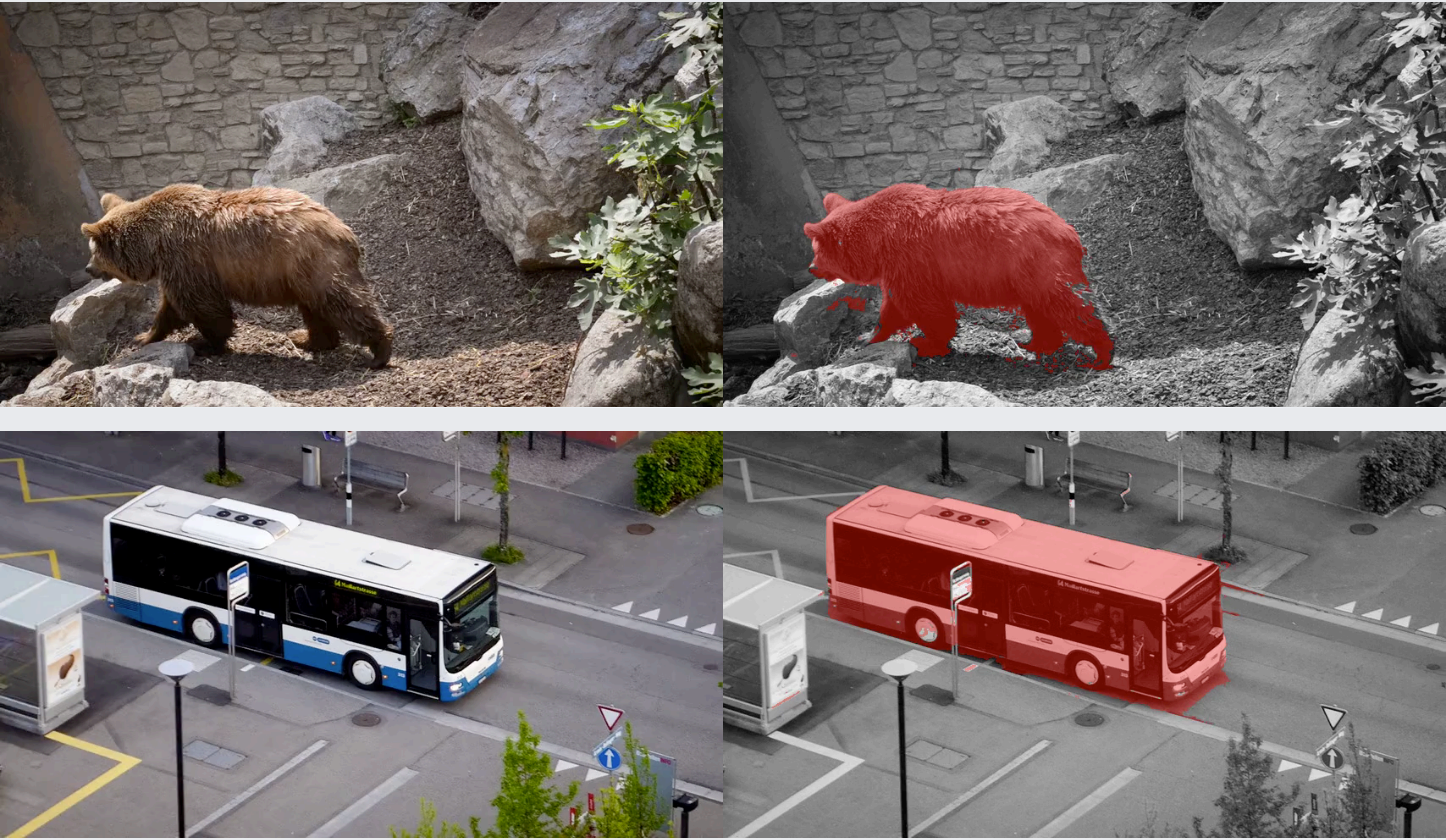
marmot

Sat, Jul 16, 2016

35

Image Search





Sage Chapel
147 Ho Plaza

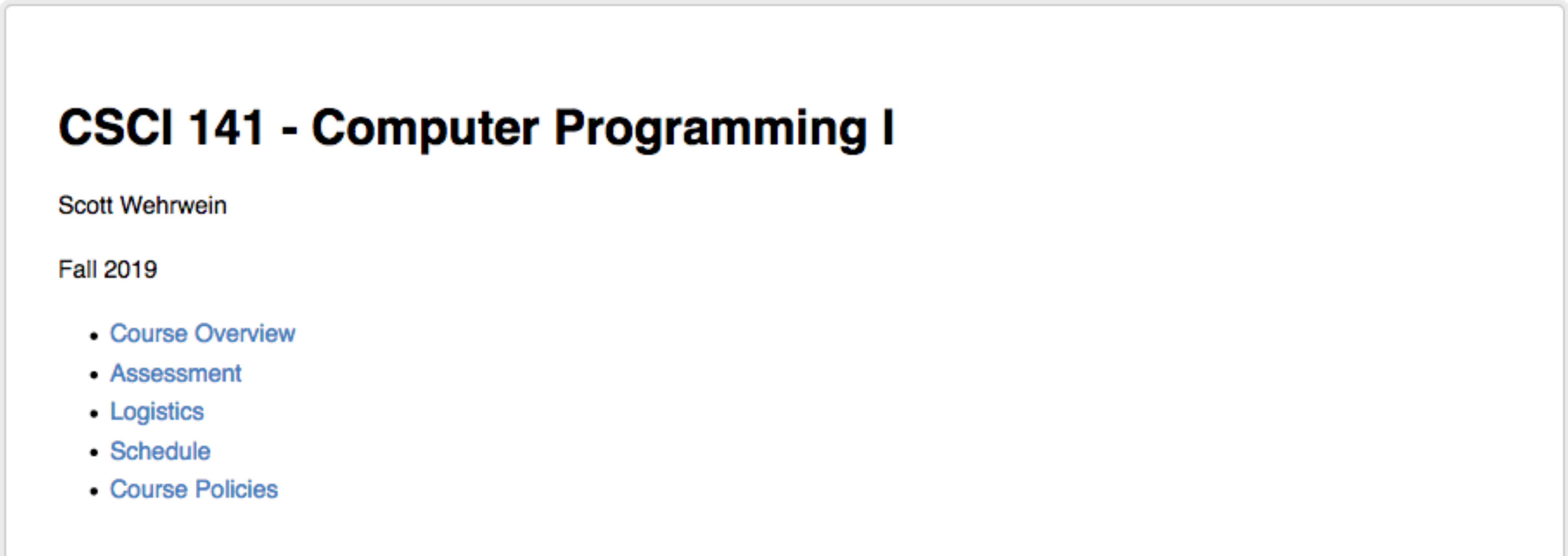
Sage Chapel
147 Ho Plaza

# Logistics

The syllabus is [on] the course webpage:

https://facultyweb.cs.wwu.edu/~wehrwes/courses/csci141_19f/

**This link can also be found on the Syllabus page on Canvas.**



## CSCI 141 - Computer Programming I

Scott Wehrwein

Fall 2019

- Course Overview
- Assessment
- Logistics
- Schedule
- Course Policies

# Syllabus Highlights

- Grade components: labs, assignments, formative, exams
- QOTD / Poll Questions
  - no credit for missed/late polls/qotd. 3 qotd and 9 polls dropped.
- Labs - attendance required
- Schedule
- Slip days
- Academic honesty

# Info overload!

Image: https://www.projectmanager.com/blog/prevent-information-overload

Some immediate implications for you:

- Read and bookmark the syllabus (find it via the canvas Syllabus page).
- Bring a device (smartphone, laptop, etc.) to every class starting Friday.
  - If you don't have a device, email me - ATUS has devices you can borrow.
- Also bring scratch paper and a writing implement.

# About You

Today's QOTD: About you!

- Q1: How many quarters have you been at Western?
- Q2: How many months of programming experience do you have?
- Q3: Are you planning to major in CS?
- Q4: Why do you want to take this class?
- Q5: Name one activity you enjoy outside of school.

Submit your answers to the **QOTD 9/25** quiz on Canvas before Friday's lecture*.

# About You

Today's QOTD: About you!

- Q1: How many quarters have you been at Western?
- Q2: How many months of programming experience do you have?
- Q3: Are you planning to major in CS?
- Q4: Why do you want to take this class?
- Q5: Name one activity you enjoy outside of school.

Submit your answers to the **QOTD 9/25** quiz on Canvas before Friday's lecture*.

* By Thursday night would be ideal

# My Expectations

Q2: How many months of programming experience do you have?

# My Expectations

Q2: How many months of programming experience do you have?

0

# My Expectations

Q2: How many months of programming experience do you have?

0

Some of you have prior experience, but zero is what I will assume.

# My Expectations

Q2: How many months of programming experience do you have?

0

Some of you have prior experience, but zero is what I will assume.
We'll look at the survey results and talk more about this next time.

# Let’s write some code already

# Let’s write some code already

- **Python** is our chosen programming language in this course.

# Let’s write some code already

- **Python** is our chosen programming language in this course.

- A programming language is a language a computer can “understand” and execute (more on what this means next time)

# Let’s write some code already

- **Python** is our chosen programming language in this course.
- A programming language is a language a computer can “understand” and execute (more on what this means next time)
- We’ll use a program called **Thonny** to write our Python code.

# Let’s write some code already

- **Python** is our chosen programming language in this course.
- A programming language is a language a computer can “understand” and execute (more on what this means next time)
- We’ll use a program called **Thonny** to write our Python code.
- Thonny is an example of an “Integrated Development Environment” (IDE): a program that provides all the features you need to write, run, and fix errors in programs.

# Without further ado...

Hello, world!

# Hello, world!

- hello.py
- Concepts demonstrated:
  - Basic usage of Thonny
  - Comments
  - Print function
  - Single and double quoted strings