

---

---

# Computer Science

Summer Springboard

---

---

# Day 1 - Introductions



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

# Allow me to introduce myself...

- Professor introduction

# Your turn!

- Name?
- Hometown?
- Favorite book / movie / show?
- What are you interested in studying?

# Now, let's introduce Computer Science!

# What is Computer Science?

- Computer Science is the study of computers and computational systems. This includes their design, development, operation, and application. Computer Science encompasses a wide range of topics and areas of research pertaining to the use of computers and computational techniques to solve problems and process information.



## What is Computer Science (cont'd)

- There is a lot to break down in that definition!
- However, the key takeaway is...

# Problem Solving!



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

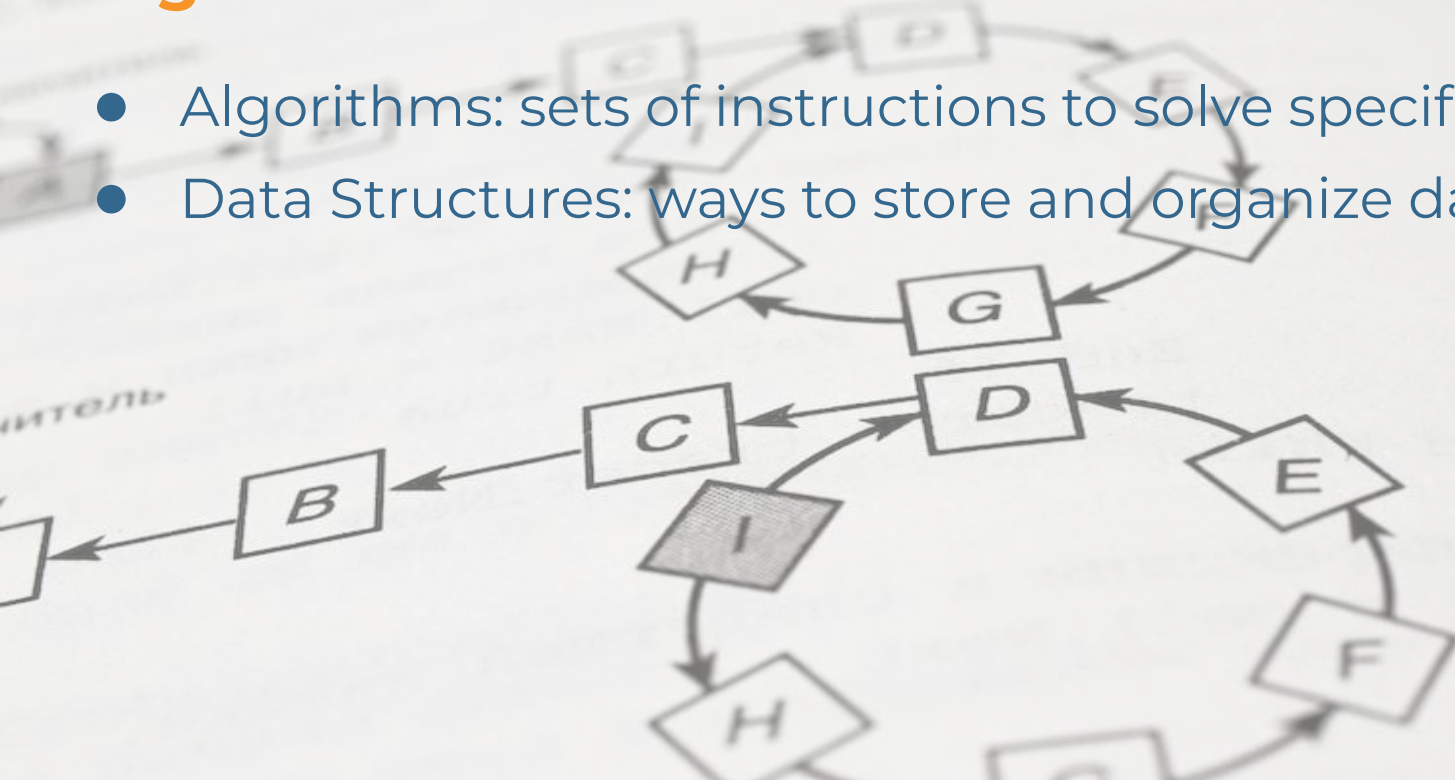


# What is Computer Science? (cont'd)

- Computer Science is all about solving problems
- Not just solving problems, but solving them to the best of our ability
- Coding is an important part of this puzzle, but not everything
- You may be interested in...

# Algorithms and Data Structures

- Algorithms: sets of instructions to solve specific problems
- Data Structures: ways to store and organize data efficiently



# Cybersecurity

- The practice of protecting systems, networks, and programs from digital attacks. This includes cryptography, network security, information security, etc.

# Human-Computer Interaction

- This field focuses on the design and use of computer technology, particularly the interfaces between users (humans) and computers

## And so much more!

- Computer Science is such a vast field that there is a niche for everyone
- For the time being, let's focus on coding

# So what's coding?

- Coding is the process of creating instructions for computers using programming languages. It is a fundamental aspect of developing software, applications, websites, etc etc etc.

# Coding Breakdown! Coding involves...

- Writing code (of course): at its core, coding involves writing the lines of code that tell a computer what to do using a programming language. Each language has its own syntax and specific use cases.
- Problem solving (here it is again): programmers use code to create algorithms and solutions to specific tasks or challenges, whether it be calculating numbers, sorting data, automating tasks, ... the list goes on.
- Creativity and expression: coding allows for creative expression (think video games, digital art, interactive websites).
- Continuous learning and adaptation: coding is always changing. There are always new languages, frameworks, and technologies. Coders have to be able to learn and adapt to keep up with the latest developments.

# Okay, Computer Science is big and important blah blah blah. But what's our goal?

- Our goal is to get our feet wet in the world of computer science, specifically through coding.
- We will be learning the basic ins and outs of a programming language that is becoming increasingly important and popular...



# Python!

- Wait... not THAT type of python.



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

# Python!



# Why Python?

- There are so many programming languages to choose from
  - C / C++
  - Java
  - Haskell
  - F#
  - Rust
  - Dare we mention Assembly?
- So why are we using Python?

# Why Python? (Cont'd)

- Python is beginner-friendly
  - Clean, readable syntax
  - Indentation enforces consistent, pretty code
- Python is versatile
  - Machine learning
  - Scientific computing
  - Scripting
  - So, so, so much more
- Python is popular
  - Large, active community of helpful developers
  - Abundance of libraries and frameworks



# So where did Python come from anyways?

- Development on Python began in the late 1980s when Guido van Rossum started working on it during his Christmas break.
- His goal was to create a language that would overcome the limitations of ABC, a language he worked on previously.
- Python (said to be named after *Monty Python's Flying Circus*) would publicly launch in February of 1991.

# Ready?

- If you are ready, let's begin!

# Finite Automata

“Wait that’s not Python!!!”

- Yes, the astute among you may notice that the title of this slide is not Python. Nonetheless, this is a great place to start.

# Finite Automata (Cont'd)

- A finite automaton is a computational model used in computer science to represent and recognize patterns in strings (some sequence of symbols e.g. this sentence).
- Although these are not particularly relevant to everyday use of Python, there are many reasons that studying them will be helpful for us.



# Why finite automata?

Learning about finite automata will help...

- Our algorithmic thinking: understanding finite automata involves breaking down problems into smaller, well-defined steps.
- Pattern recognition: finite automata are often used to recognize patterns or sequences in strings. This will be useful in Python test processing, parsing, and data validation.
- Introduce us to theoretical concepts.
- Our understanding of state machines: it is crucial to understand state machines when designing complex systems or simulations in Python, such as game development or control systems.

# Alright, so what is a finite automata anyways?

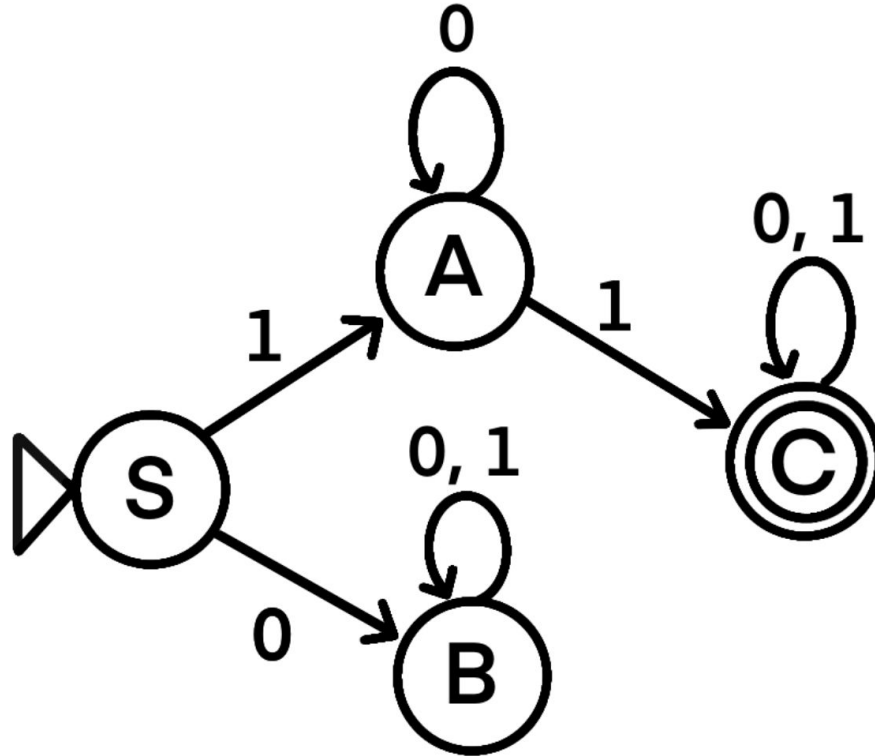
- A Deterministic Finite Automaton is a tuple  $\{Q, \Sigma, q_0, F, \delta\}$  where:
  - $Q$  is the (finite) state of states
  - $\Sigma$  is the alphabet (finite set of symbols used as input)
  - $q_0$  is the start state
  - $F$  is the set of final states
  - And  $\delta$  is the map  $Q \times \Sigma \rightarrow Q$

# That's too scary!!!

What does all of that even mean?

- Long story short: don't worry about it for now
- Without getting too deep into the nitty-gritty, we will be working with some simple deterministic finite automata (DFAs)
- For now, just think of a DFA as a special kind of game that follows specific rules to decide if something is true or not (visual on the next slide)

# DFA Visualized



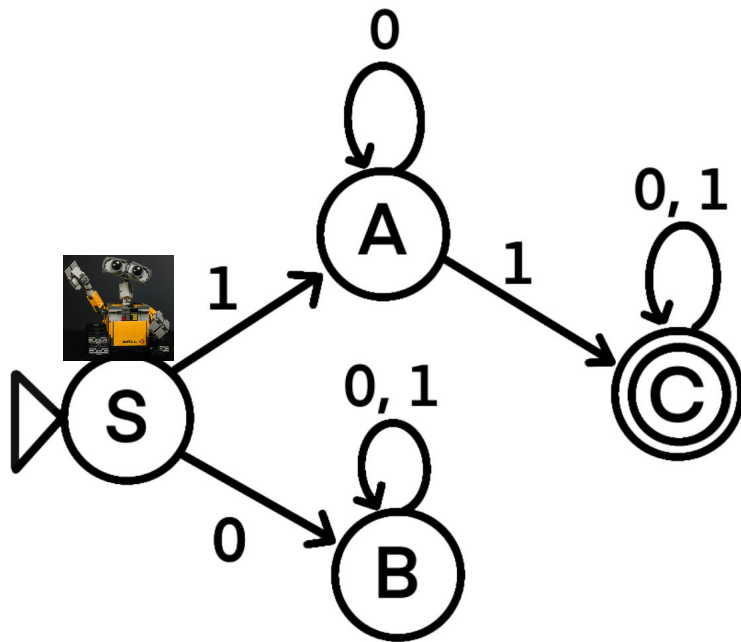
# DFA Explained

- Imagine you have a robot that lives in a world with a few rooms connected by doors. Each room has a rule about which door the robot can go through next, based on a secret code you give it, like a series of yes or no questions. Here is how the game works:



## DFA Explained (Cont'd)

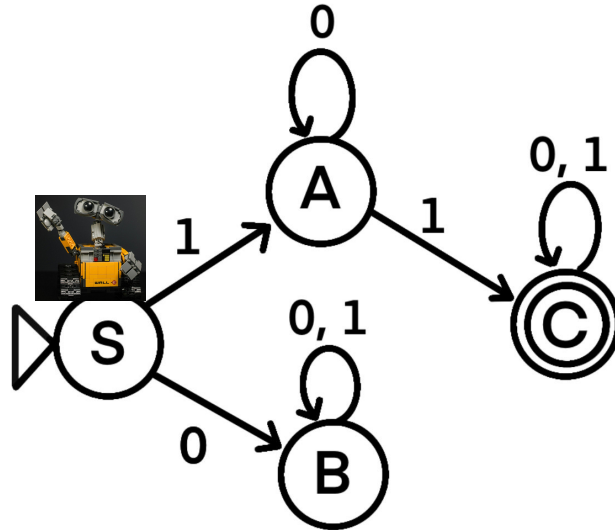
- Starting room: The robot starts in a special room, which is its home. This is the circle that has a triangle on its side



# DFA Explained (Cont'd)

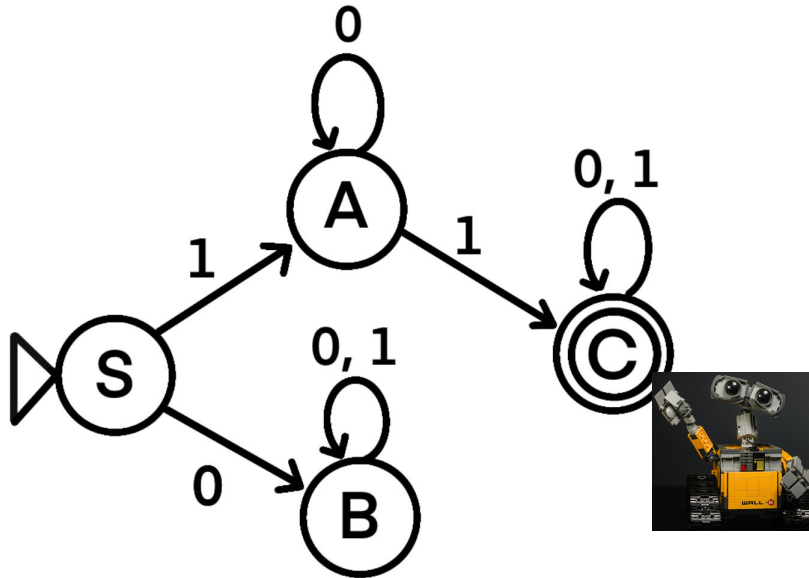
- Secret code: You give the robot a secret code made of simple stamps, like: “Step 1: Say ‘1’, Step 2: Say ‘0’”, and so on.
- Moving around: depending on each step of your secret code, the robot moves from one room to another. Each room tells the robot where to go next when it hears ‘1’ or ‘0’.

For example, if we tell the robot “1” first, it will go from S to A, but if we tell it “0” first, it will go from S to B



# DFA Explained (Cont'd)

- End of the game: When the secret code ends, the robot stops moving. If it stops in a special room (denoted by a circle inside a circle), it means your secret code wins the game! If it stops in any other room, the code doesn't win.
  - For example, if we say "1 1 0", the robot will go from S to A and finally to C, which is a special room, so 110 wins the game.

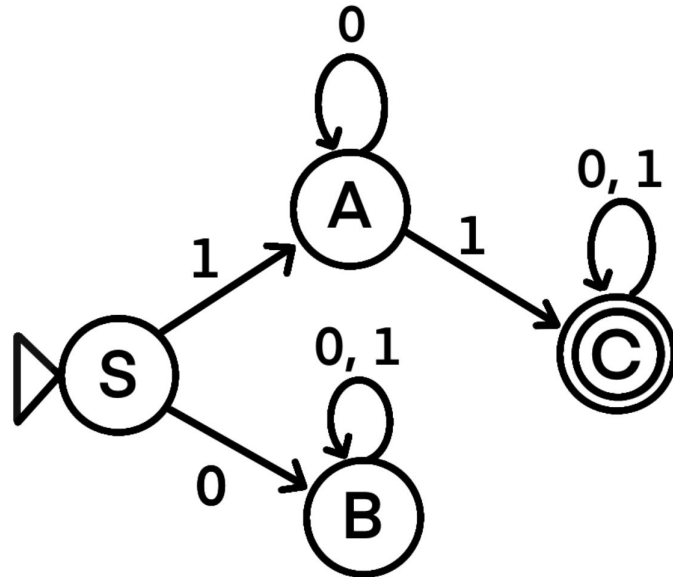


Happy  
robot!



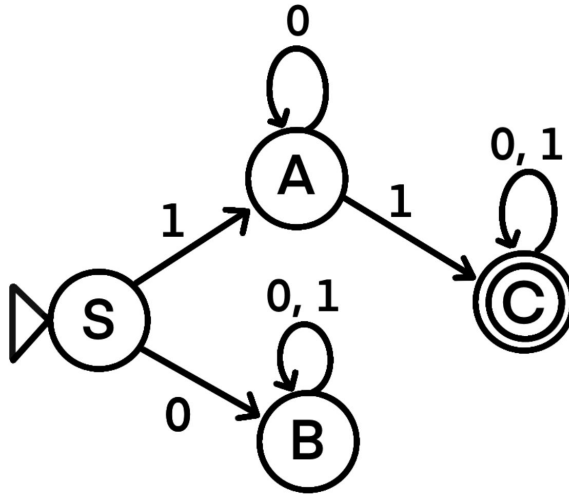
# That's not so bad!

- Seems simple enough, right?
- So, what are all of the secret codes that will help us win the game?
  - 110 works
  - 101 works
  - 1111111111111111 works
  - What else works?
- Think about this for a few minutes with a partner.



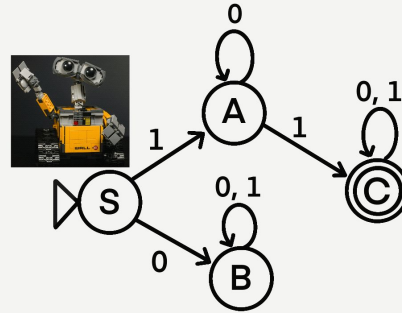
# The Answer

- Well, it turns out that an infinite amount of codes will work.
  - But not every code; i.e. 0 0 0 does not work
  - So, the answer is that any code that starts with a 1 and has at least one other 1 later in the code will work.



# Let's see this animated with the code 100111

Notice that when we reach C, our code continues; we just keep looping back to C.



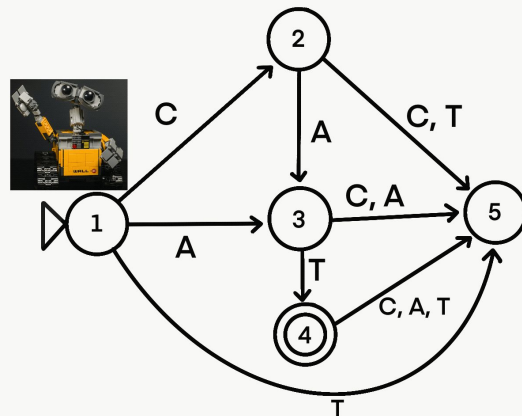
## One more example

- Let's say we have a DFA which uses the letters {C, A, T}
- Since the robot follows very specific directions, each room that it is in needs to have an instruction for if the robot receives any given letter. For example, if the robot is in room 2, it needs to know what room to go to if it is given 'C', if it is given 'A', and if it is given 'T'.
- Let's see what a DFA would look like with the secret codes "CAT" and "AT".

# {C, A, T} DFA Animated

Notice how we win if we enter a winning code like CAT, but if we enter a losing code like CATT, we lose. The only two winning codes here are CAT and AT. No other code will have us end on our special room, 4.

Note: This is certainly not the only way to draw this DFA! There are other valid ways to meet our specifications.

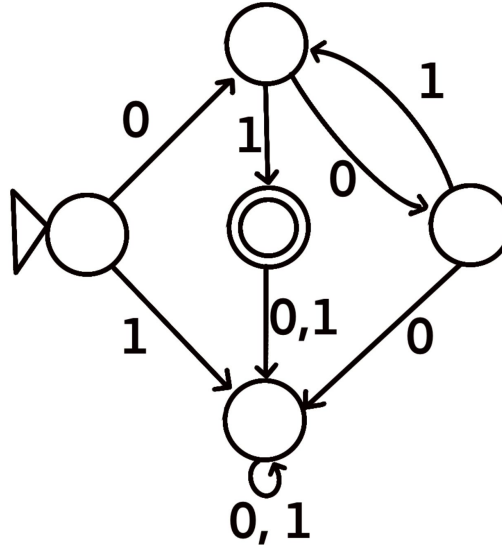


**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

# Your turn!

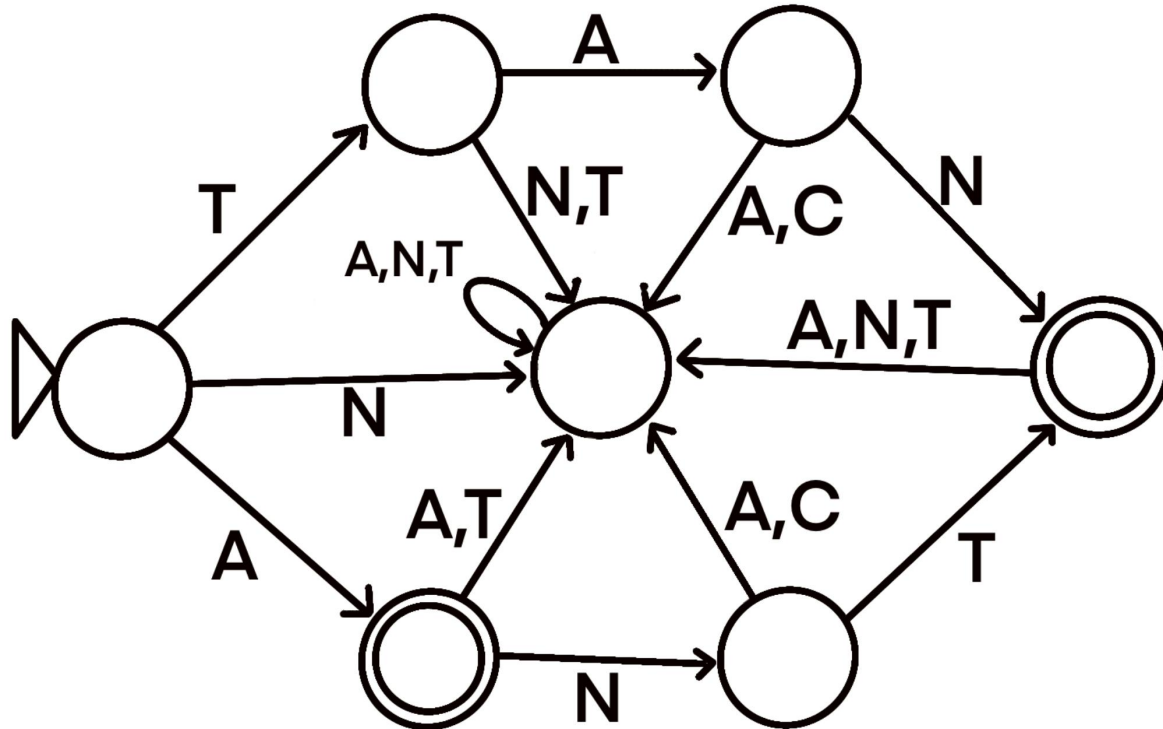
Our robot finds itself in a two new worlds and it needs your help!

1. Design a world that works with the letters {A, N, T} that has winning codes “A”, “ANT”, and “TAN”. Get help from a partner or group if needed! [20 minutes]
2. Determine all winning codes for the DFA pictured below. Once again, get help from a partner or group if needed! [15 minutes]



# Answers

1. Here is a sample solution (many possible solutions exist):



## Answers (Cont'd)

2. This DFA accepts all codes which start with a 0, then contain any number of repetitions of the sequence 01, followed lastly by a single 1.

- Examples include:
  - 01
  - 0 011
  - 0 01 01 01 01 01 01 1
- Non-examples include:
  - 0
  - 1
  - 011
  - 001
  - 0000



# DFA W(rap)up

♪♪♪Jumpin' through states like a hopscotch game,♪♪♪  
♪♪DFA's the name, determinism's the fame,♪♪  
♪♪♪Follow my rules, no guesswork in my lane!♪♪♪

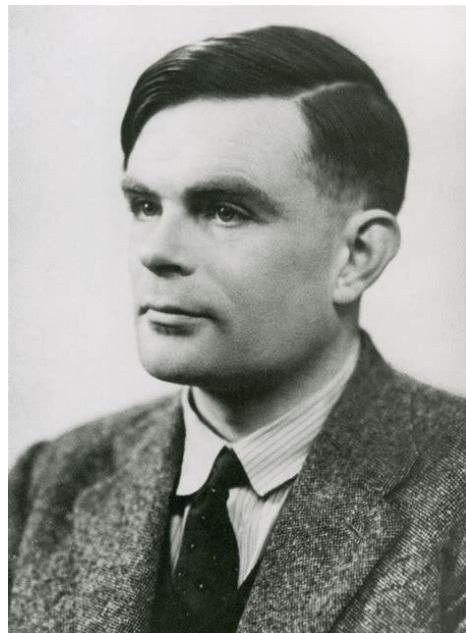
All jokes aside, those last couple of questions got complicated real fast! And our alphabet was small; imagine using the whole set of letters in the English alphabet! There is so much more to know and learn about automata, but it would take a semester-long class to even start to do this topic justice. We got what we wanted out of it for now though: we strengthened our critical thinking and problem solving skills, while getting into the mindset of a computer. When we begin coding in Python, we will see that the computer interprets things in a similar way to what we just did; it reads our code step-by-step, line-by-line.

# Want to learn more about this topic?

- [https://en.wikipedia.org/wiki/Finite-state\\_machine](https://en.wikipedia.org/wiki/Finite-state_machine)
- [https://en.wikipedia.org/wiki/Context-free\\_grammar](https://en.wikipedia.org/wiki/Context-free_grammar)
- [https://en.wikipedia.org/wiki/Turing\\_machine](https://en.wikipedia.org/wiki/Turing_machine)

Keep at this topic, and eventually you'll learn about the Turing machine!

Alan Turing, Cambridge alumnus ->



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward

# Alright, back to Python

- Most likely, your computer does not automatically have Python available and ready to use
- So, let's fix that by installing Python!

# Python Installation



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

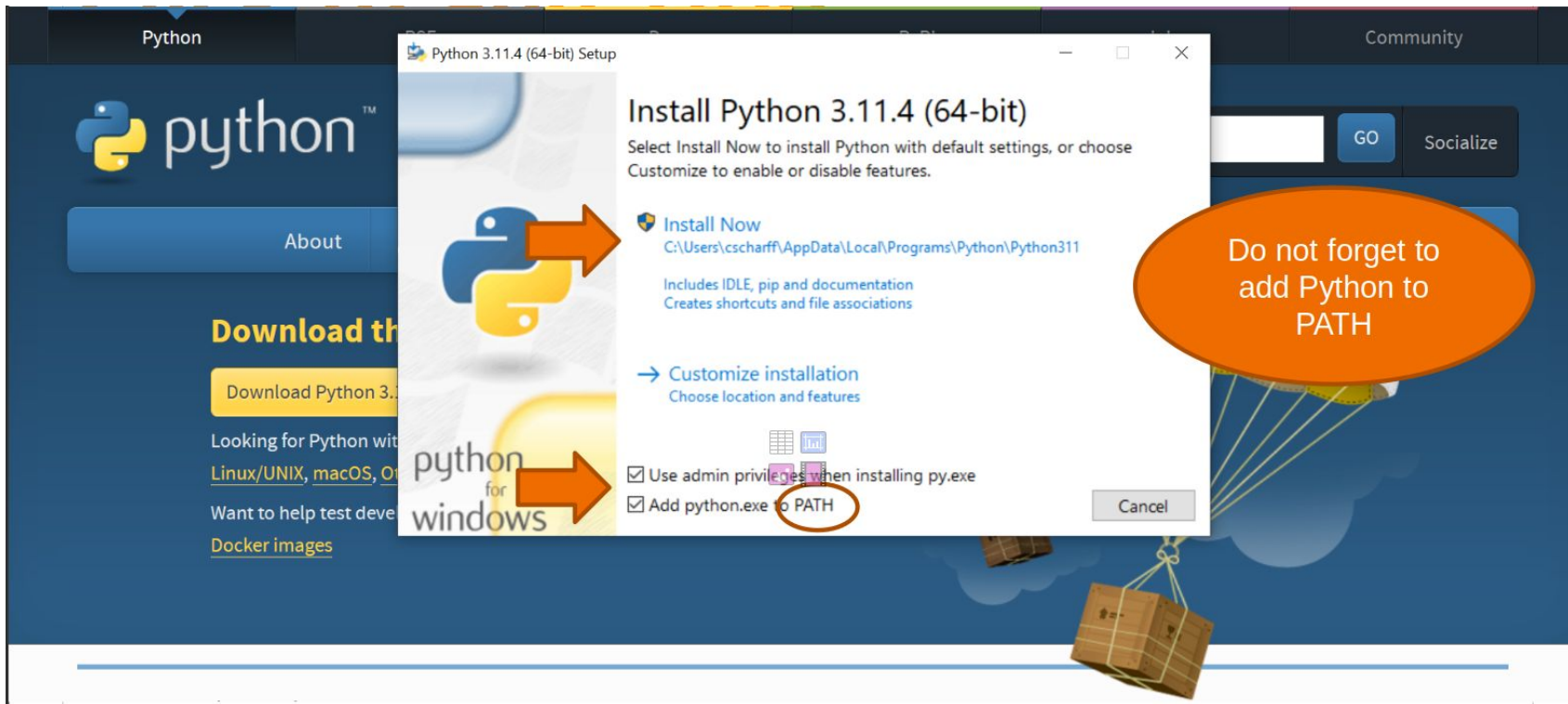
# Python Installation (Step 1)

First, navigate to <https://www.python.org/downloads/>

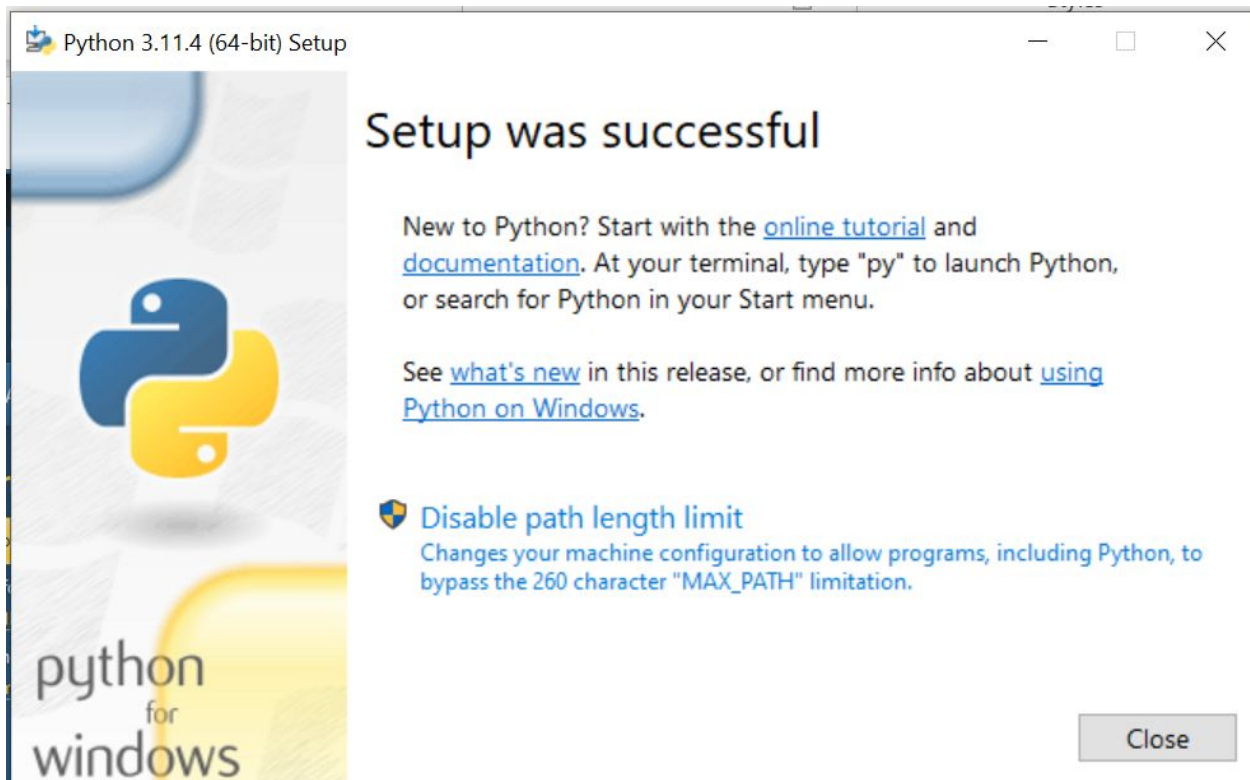


**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward

# Python Installation (Step 2)



# Python Installation (Step 3)



## Python Installation (Step 4)

Make sure that Python is installed by opening your terminal (type 'terminal' into the search bar) and type either:

“python3 --version” or “python --version” and press enter.

You should then see something like this show up:

“Python 3.10.12”



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward



# IDE's

- We could write .py files directly and run them through the command line, but that is a lot of work.
- Instead, let's use an Integrated Development Environment, which is an application that programmers use to write and test their code. It is like a toolbox for coding, combining functions like code editing, error checking, running / testing code, and much more!

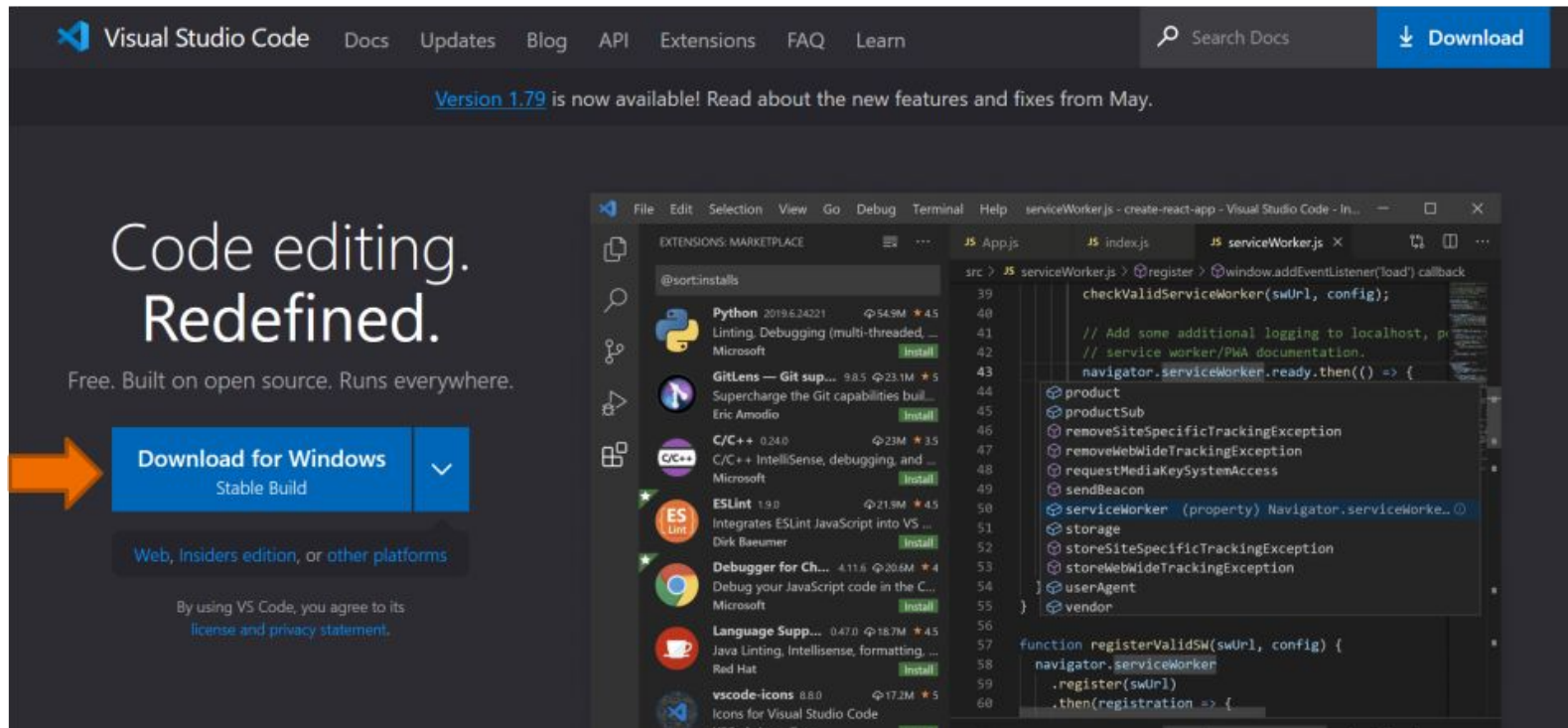
# Visual Studio Code and Google Colab

- Visual Studio (VS) Code: super popular, open-source IDE from Microsoft. It is versatile, has extensive plugin support, and strong integration with various programming languages and tools.
- Google Colab: free, cloud-based platform that allows you to write and run Python code. It is primarily used for education, data analysis, and machine learning.

# IDE Setup

- We will install VS Code so that you can use it on your own time. It is very nice and will be very useful when you are more advanced.
- For now, we will use Google Colab for simplicity.

# Download Visual Studio Code



Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Search Docs Download

Version 1.79 is now available! Read about the new features and fixes from May.

## Code editing. Redefined.

Free. Built on open source. Runs everywhere.

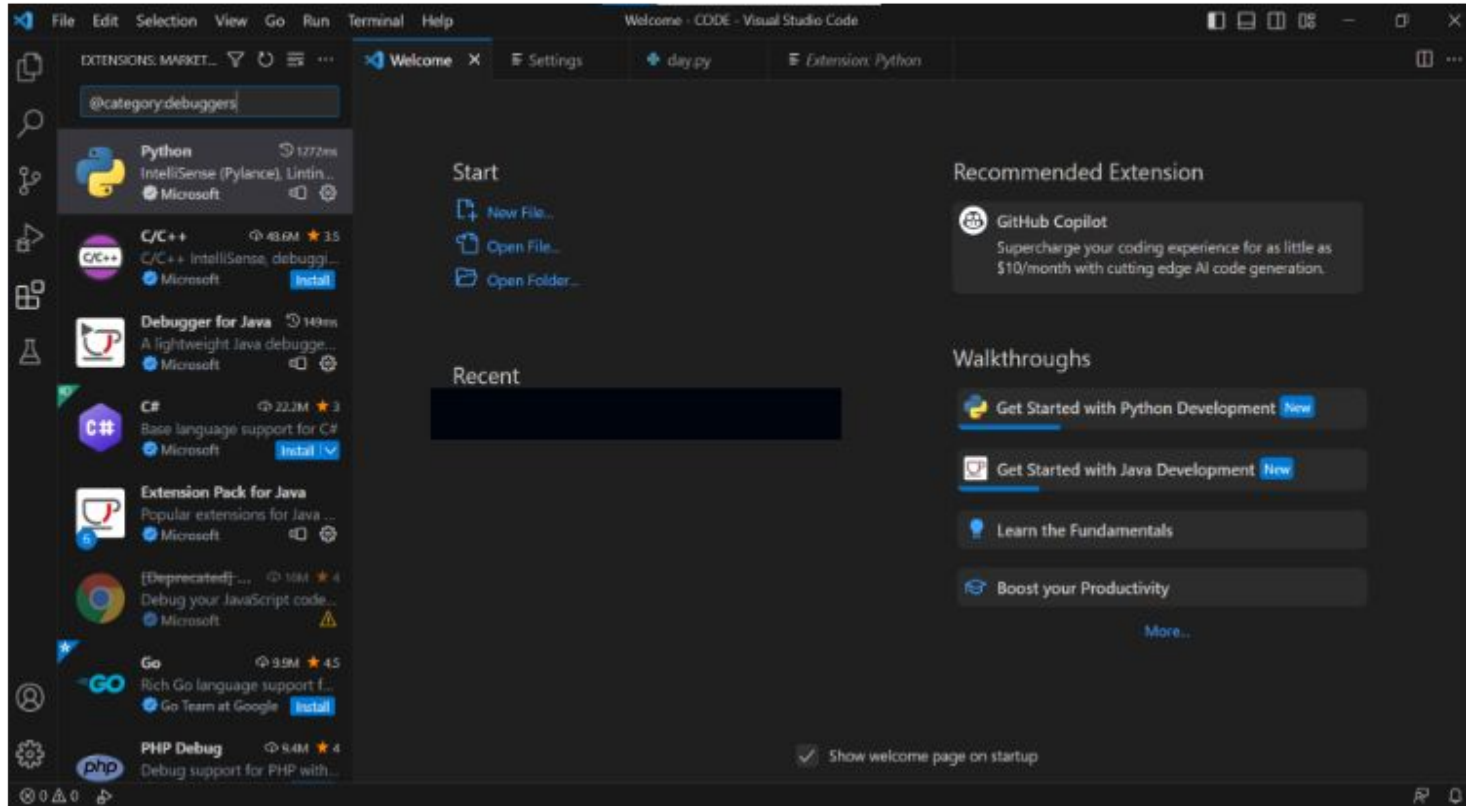
**Download for Windows**  
Stable Build

Web, Insiders edition, or other platforms

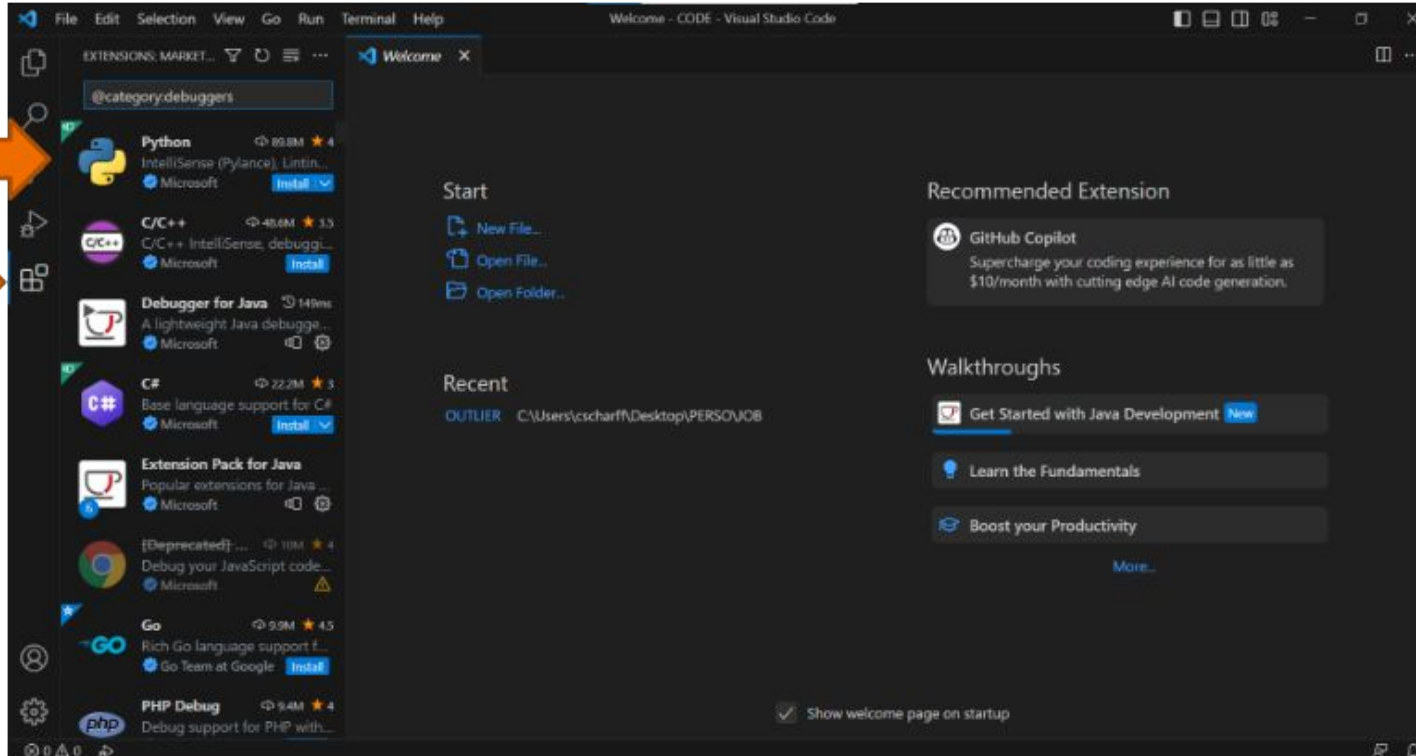
By using VS Code, you agree to its [license and privacy statement](#).

The screenshot shows the Visual Studio Code interface with the Extensions Marketplace on the left, listing various extensions like Python, GitLens, C/C++, ESLint, and the Debugger for Chrome. The main editor area shows a JavaScript file named 'serviceWorker.js' with code for registering a service worker and handling events.

# Launch Visual Studio Code



# Install the Python Extension





# Python v2023.10.1

Microsoft [microsoft.com](https://microsoft.com) | 89,846,507 | ★★★★★ (545)

IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebooks, code f...

Installing



Be patient!



# Python v2023.10.1

Microsoft [microsoft.com](https://microsoft.com) | 89,846,507 | ★★★★★ (545)

IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebooks, code f...

Disable

Uninstall

Switch to Pre-Release Version

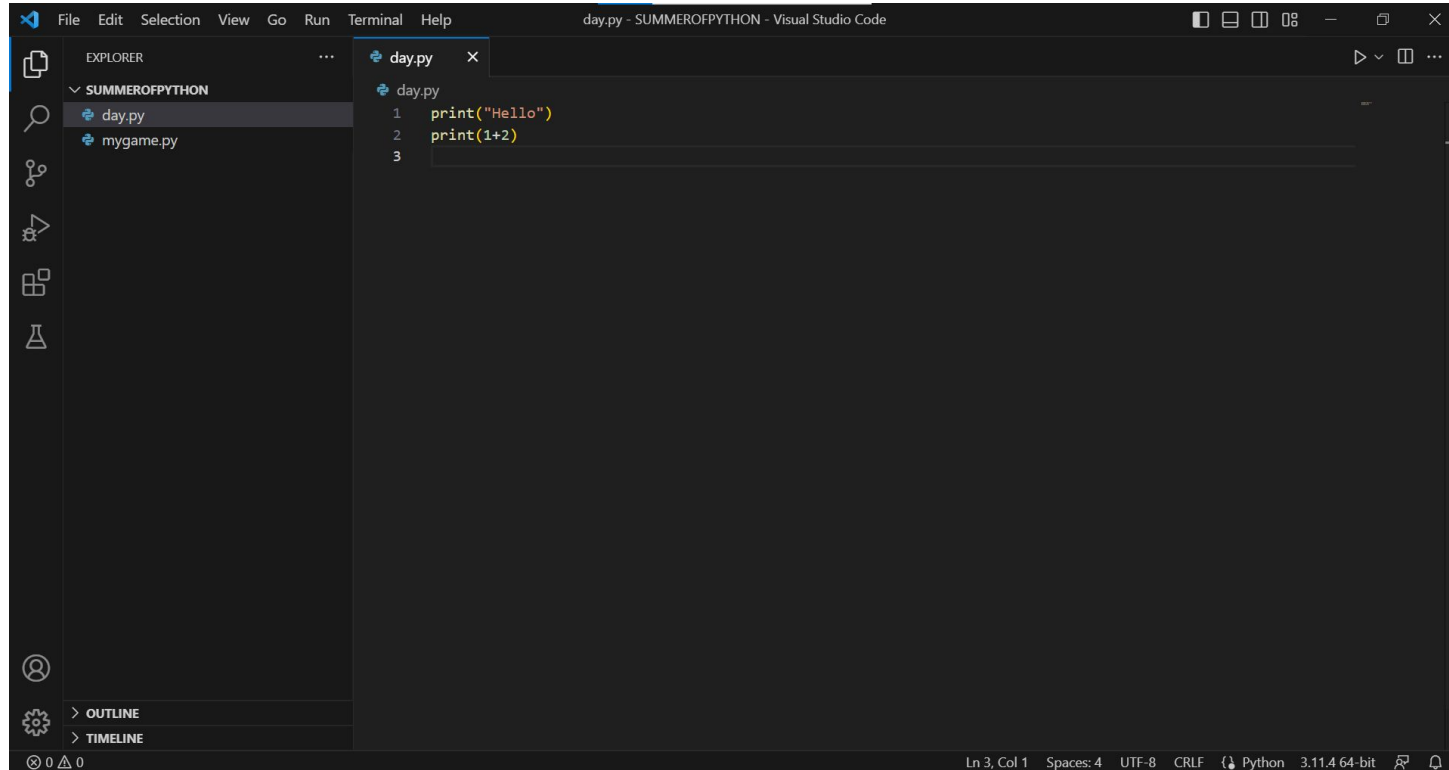


This extension is enabled globally.



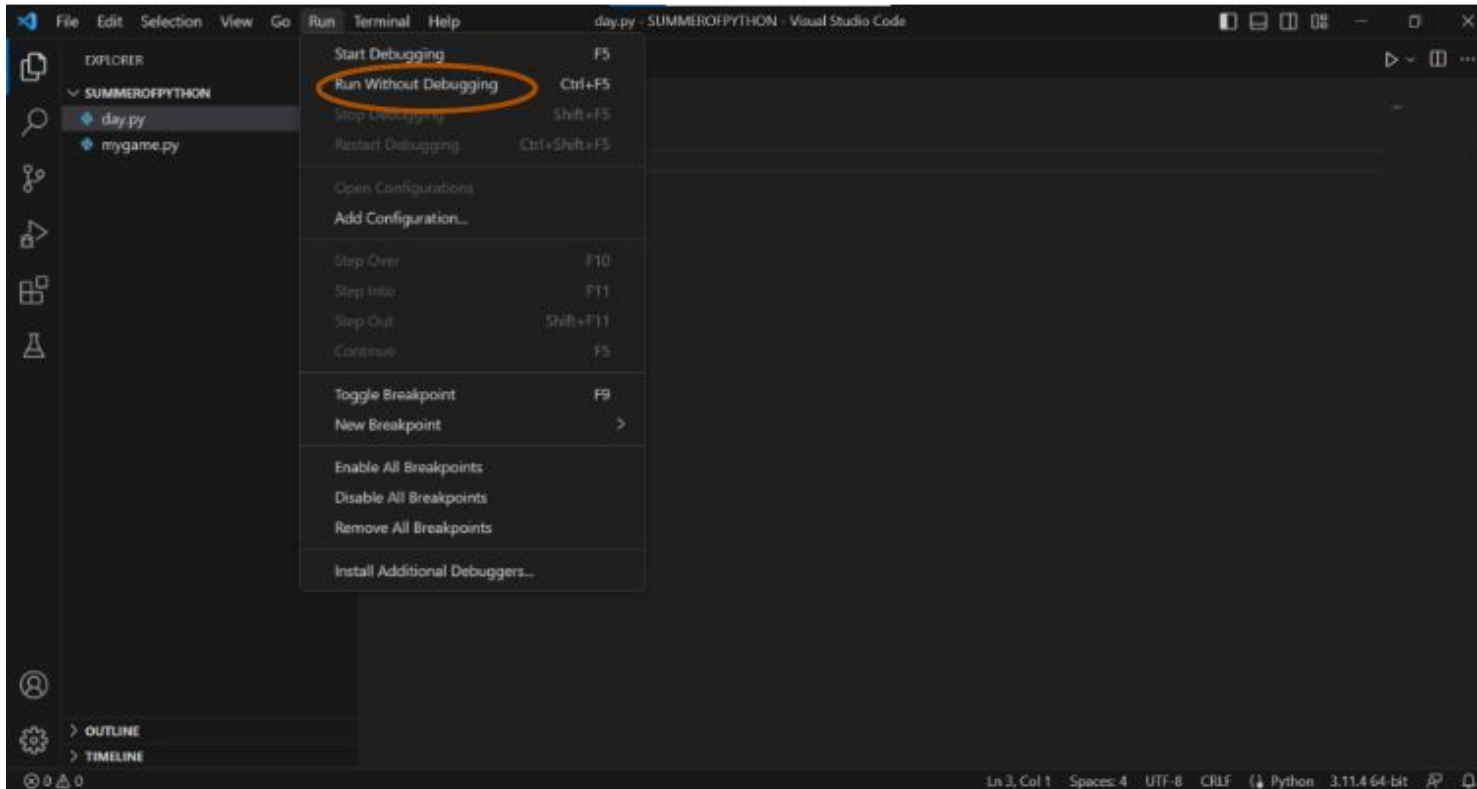
**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward

# Create a folder on your computer, open it and start coding

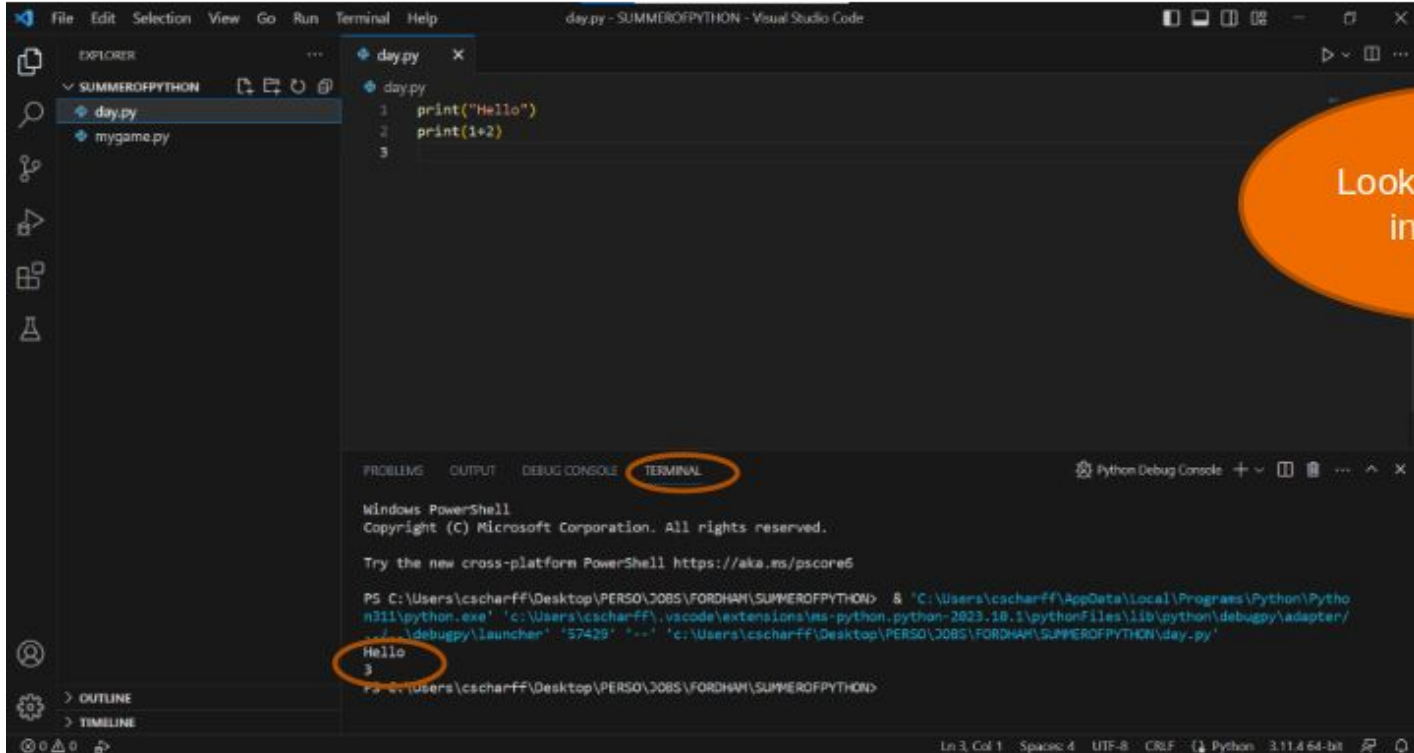




# Run the code



# Run the code



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a project named SUMMEROFPYTHON with two files: day.py and mygame.py. The day.py file is open in the editor, showing the following code:

```
1 print("Hello")
2 print(1+2)
3
```

The TERMINAL pane at the bottom shows the output of running the code:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

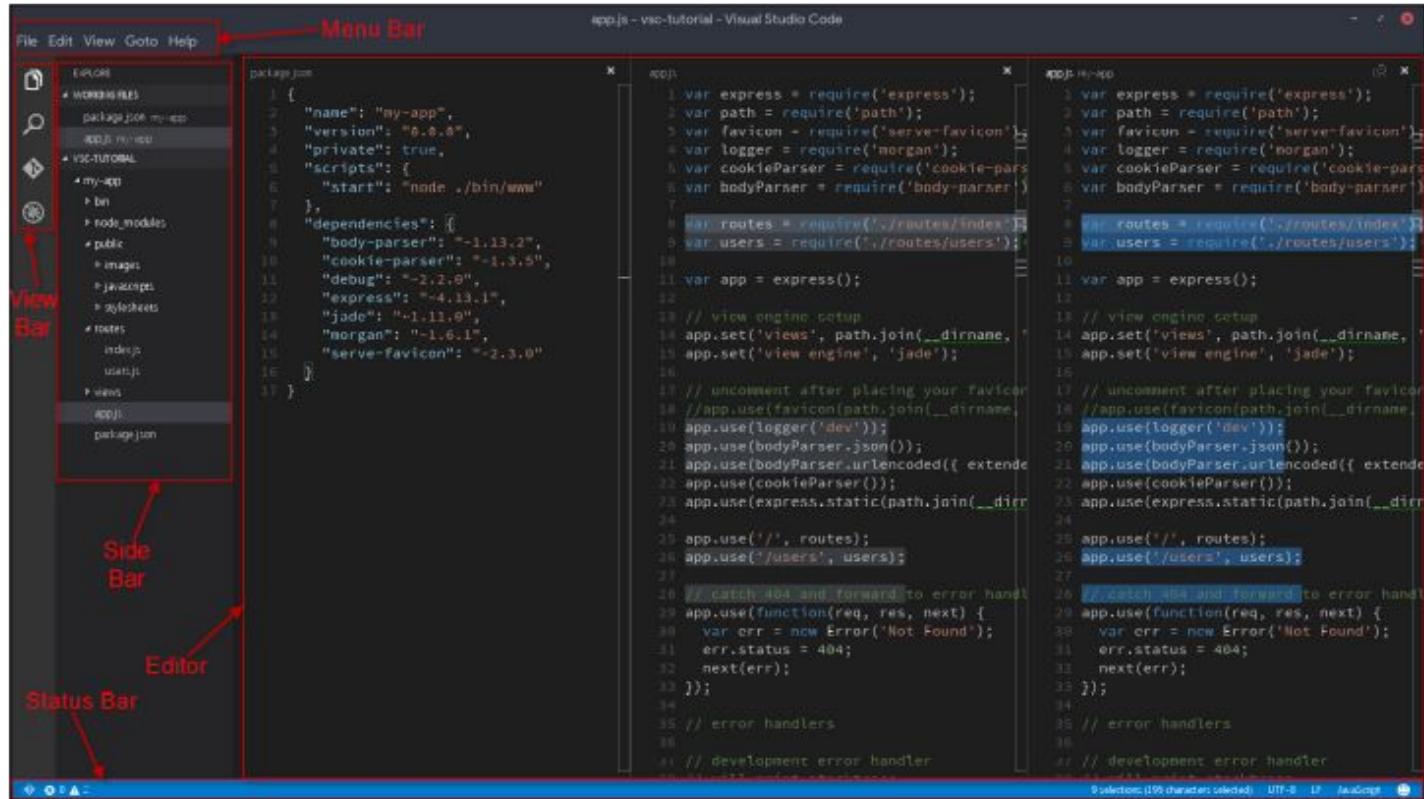
PS C:\Users\cscharff\Desktop\PERSON\JOBS\FORDHAM\SUMMEROFPYTHON> & 'C:\Users\cscharff\AppData\Local\Programs\Python\Python311\python.exe' 'C:\Users\cscharff\.vscode\extensions\ms-python.python-2023.18.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57429' '--' 'c:\Users\cscharff\Desktop\PERSON\JOBS\FORDHAM\SUMMEROFPYTHON\day.py'

Hello
3
PS C:\Users\cscharff\Desktop\PERSON\JOBS\FORDHAM\SUMMEROFPYTHON>
```

An orange oval highlights the output "Hello" and "3" in the terminal.

Look at the result  
in Terminal

# Overview of Visual Studio Code



# Google Colab

- <https://colab.research.google.com/>
- The cool part is: no setup and no downloads are needed!  
Just navigate to this website and you are set.

# Time to Code!



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.

# Quick Summary

- Before we jump over to today's notebook, lets cover a few important topics:
  - Comments
  - Print
  - Binary Operators

# Comments

- A comment is a piece of text in your code that is not executed by the computer
- It's used to leave notes for yourself or others to explain what the code is doing. In Python, you can make a comment by putting a hash symbol (#) at the start of the line.
- Everything after the # on that line is treated as a comment and ignored by Python when running the code.
  - For example, # This is a comment in Python.

# Print

- Print is a function in Python that is used to output data to the console, make it visible to the user.
- To use 'print', type 'print()' and put what you want to display inside the parentheses.
  - For example, 'print("Hello World")' will display the text "Hello World"



# Binary Operators

- Binary operators in Python are used to perform operations on two operands (values or variables).
  - Examples: + (addition), - (subtraction), \* (multiplication), and / (division).
- To use a binary operator, place it between two values or variables
  - For example: 'a + b' adds the values of 'a' and 'b'

## To the notebook!

- Now it is time to open today's Google Colab notebook. Navigate to the notebook and follow the instructions.

---

---

# Computer Science

Summer Springboard

---

---

# Day 2: Basics



**SUMMER**  
SPRINGBOARD  
Look Inward. Go Upward.