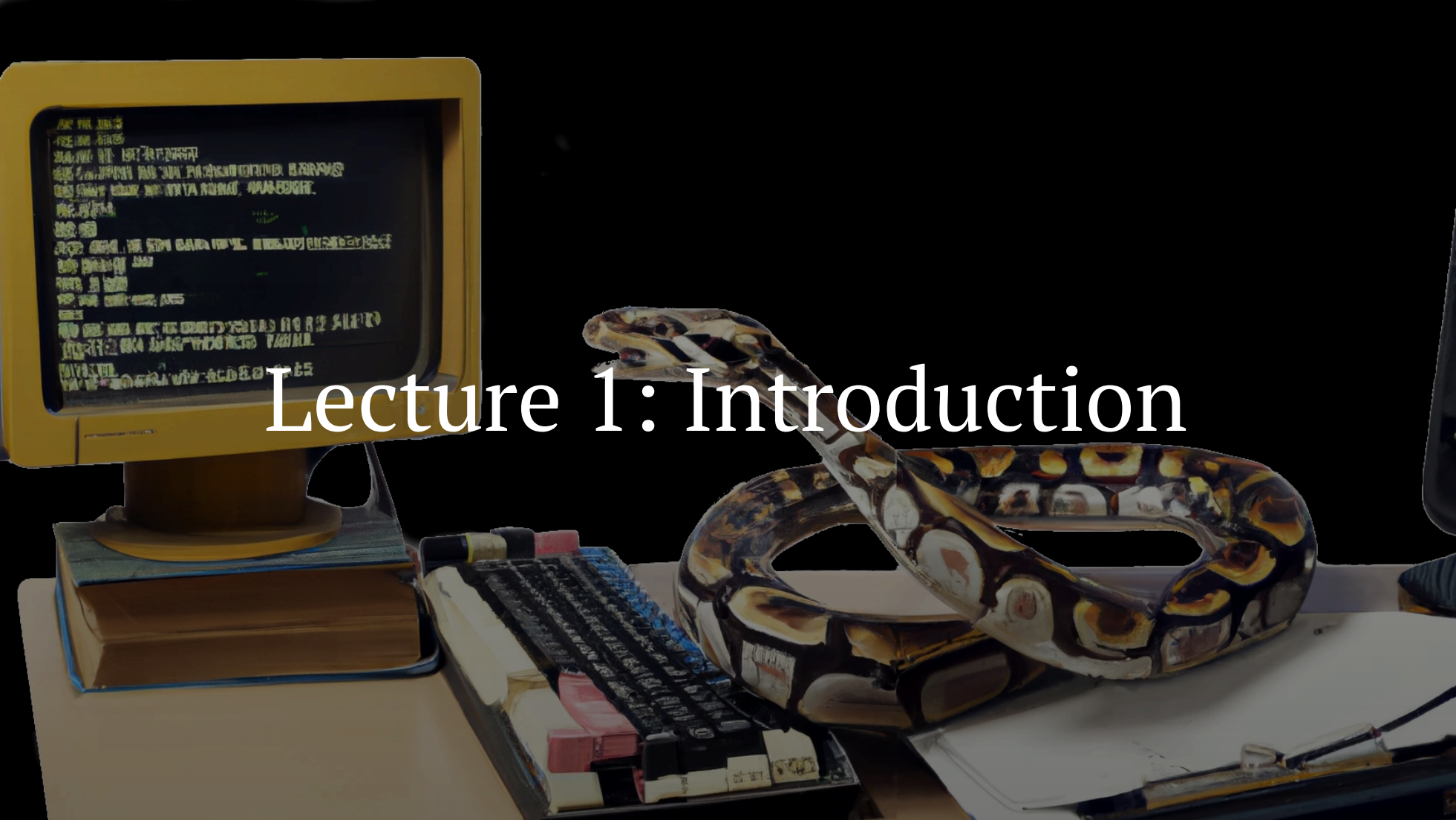


Lecture 1: Introduction

🏠 Roadmap

- Introduction
- Housekeeping
- A note about AI
- What do we want to do this year?
- Warm-up problems
- Homework assignment

🏠 Introduction

```
for person in each_of_us:
    person.tell_your_name_and_pronouns()
    person.tell_cool_projects_this_summer()
    person.tell_what_youre_excited_about_programming_this_year()
```

🏠 Meeting Times

- Lecture
  - In person: Thursday 10:00 am - 11:30 am
  - Attendance mandatory

🏠 Communication

- Message me on Jupiter Ed
- Or email me: steve.joiner@hybridgeacademy.org
- I'll help you over email or we can set up a Zoom call
- I'll usually respond quickly, but it may take an hour or more if I'm busy
- Late night messages – may not respond until next morning

🏠 Lecture

- Review of previous assignment
- Presentation of new material
- Individual help with project
- Presentation of next homework assignment

🏠 Grades

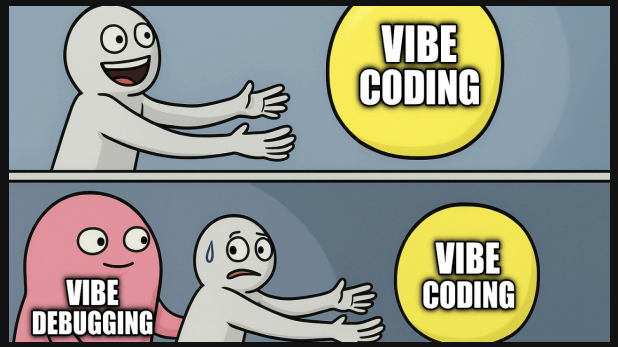
- 85% Weekly homework assignments
- 15% Attendance and participation

If we decide to do longer individual projects, then:

- 50% Weekly homework assignments
- 35% Individual project
- 15% Attendance and participation

🏠 Use of AI and Online Resources

- Do not copy/paste code from the internet
- Do not copy/paste code from AI
- Copy/paste from any source is considered cheating



- AI is everywhere
- Can be very powerful and helpful
- But can be a hindrance to learning
- 1st Semester: don’t use AI at all
- 2nd Semester: we’ll learn how to use AI effectively

- When using online resources:
  - Don’t just copy solution – understand it
  - Then write the code yourself
  - We’re here to learn, not "make the code work"
- Productive struggling is part of learning

🏠 Syllabus

- Mix of structured units and projects
- Structured units
  - Error handling
  - Use of AI in programming
  - Object-oriented programming
  - Version control (`git`)
  - Computer Science topics
    - Searching, sorting
    - Linked lists
    - Path finding
    - Binary representation
    - Compression algorithms
  - Pygame review
  - Advanced Pygame
- Projects
  - Artificial life
  - Computer vision
  - Electronics
  - Robotics
  - Godot
  - Web dev

What do *you* want to do?

🐍 Warm-up 1: Self-Referential Statement

- Write a program that prints the following:

```
This message contains ? characters.
```

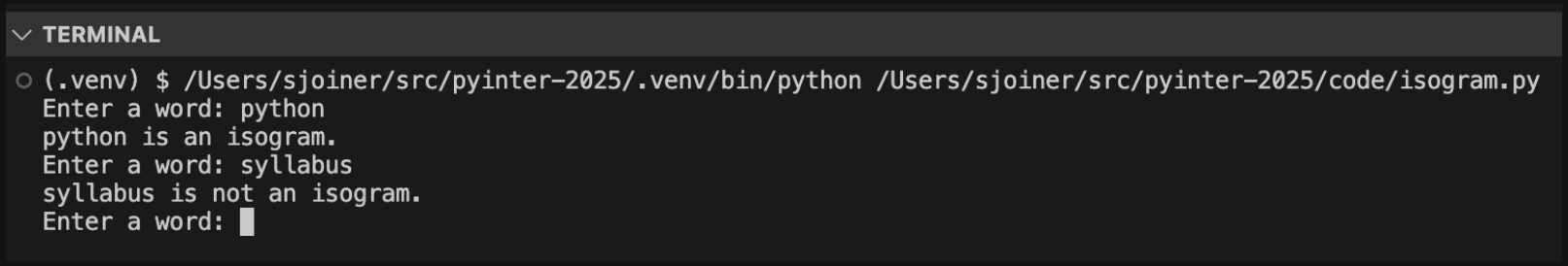
- The `?` should be replaced by a value that makes the statement true.
- Your program should figure out the value for `?`.

🐍 Self-Referential Statement Solution

```
for i in range(1000000):
    message = f"This message contains {i} characters."
    if len(message) == i:
        print(message)
```

🐍 Warm-up 2: Isograms

- Write a program that asks for a single word as input
- Then your program prints whether or not the word is an isogram
- An isogram is a word in which no letter occurs more than once



🐍 Isograms: Solution

```
while True:
    word = input("Enter a word: ")
    lcword = word.lower()

    letters = []

    for letter in lcword:
        if letter in letters:
            print(f"{word} is not an isogram.")
            break
        letters.append(letter)

    if (len(letters) == len(lcword)):
        print(f"{word} is an isogram.")
```

🐍 Warm-up 3: Brackets

- Write a program that asks for a statement
- Then the program prints wether or not the brackets in the statement are matched
- Brackets include: `(`, `)`, `[`, `]`, `{`, `}`

```
(.venv) $ /Users/sjoiner/src/pyinter-2025/.venv/bin/python /Users/sjoiner/src/pyinter-2025/code/dobracketsmatch.py
Enter a statement: abc (def) ghi
Brackets match
Enter a statement: abc {def (ghi) [jkl] mno}
Brackets match
Enter a statement: abc [def {ghi [jkl] } mno]
Brackets match
Enter a statement: abc [def {ghi [jkl] mno]
Brackets don't match
Enter a statement:
```

🐍 Brackets: Solution

```
while True:
    statement = input("Enter a statement: ")
    stack = []
    brackets = {"(": ")", "[": "]", "{": "}"}
    matched = True

    for ch in statement:
        if ch in brackets:
            stack.append(ch)
        elif ch in brackets.values():
            if len(stack) == 0 or brackets[stack.pop()] != ch:
                matched = False
                break

    if matched and len(stack) == 0:
        print("Brackets match")
    else:
        print("Brackets don't match")
```

🐍 Homework Assignment 1: Hangman

- Write a program that plays hangman with you
- The program chooses a word
- The human guesses
- Create a short list of possible words (for now)
- Show:
  - Blanks and correct guesses
  - Incorrect guesses
  - Number of remaining guesses
  - Win/lose message
- Handle invalid input and repeat guesses
- Optional: display the hangman drawing