Algorithms and Data Structures for Data Science
lab_debug

CS 277
Brad Solomon

January 27, 2023

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Department of Computer Science

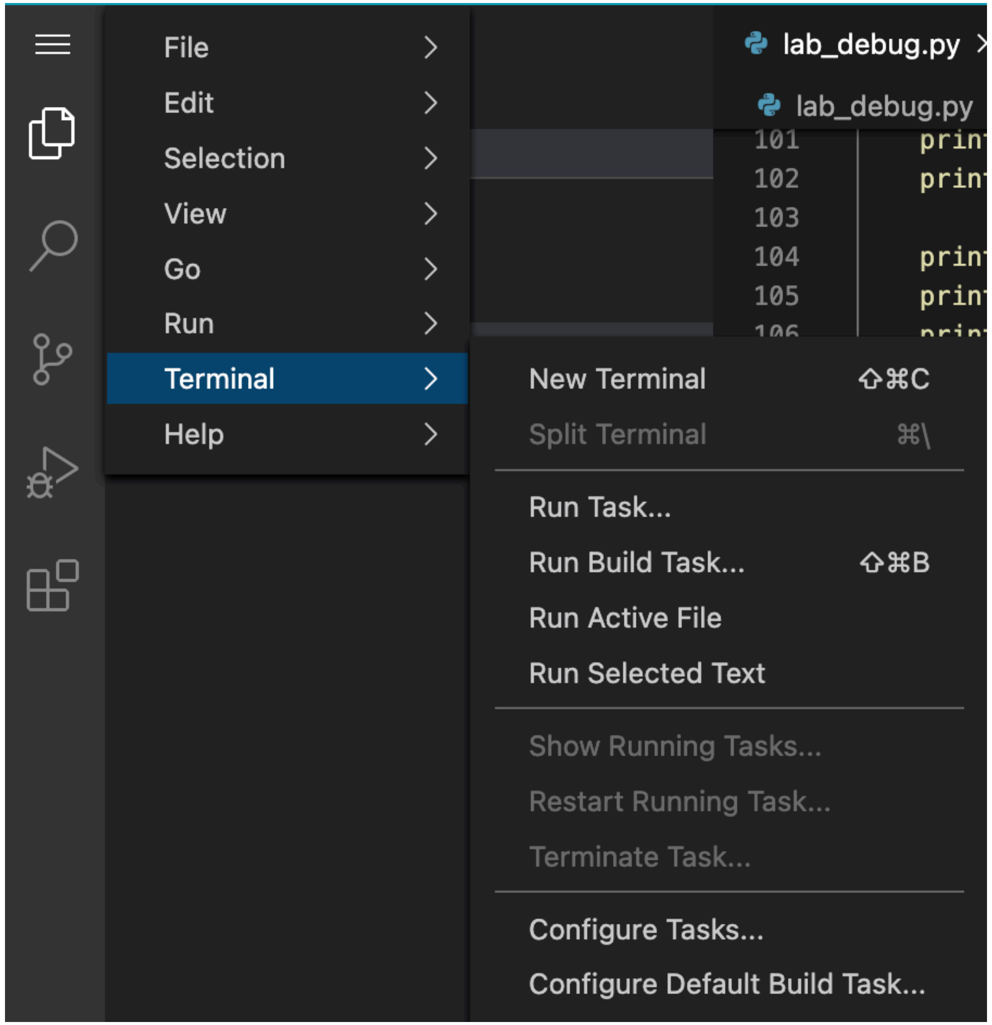
Learning Objectives

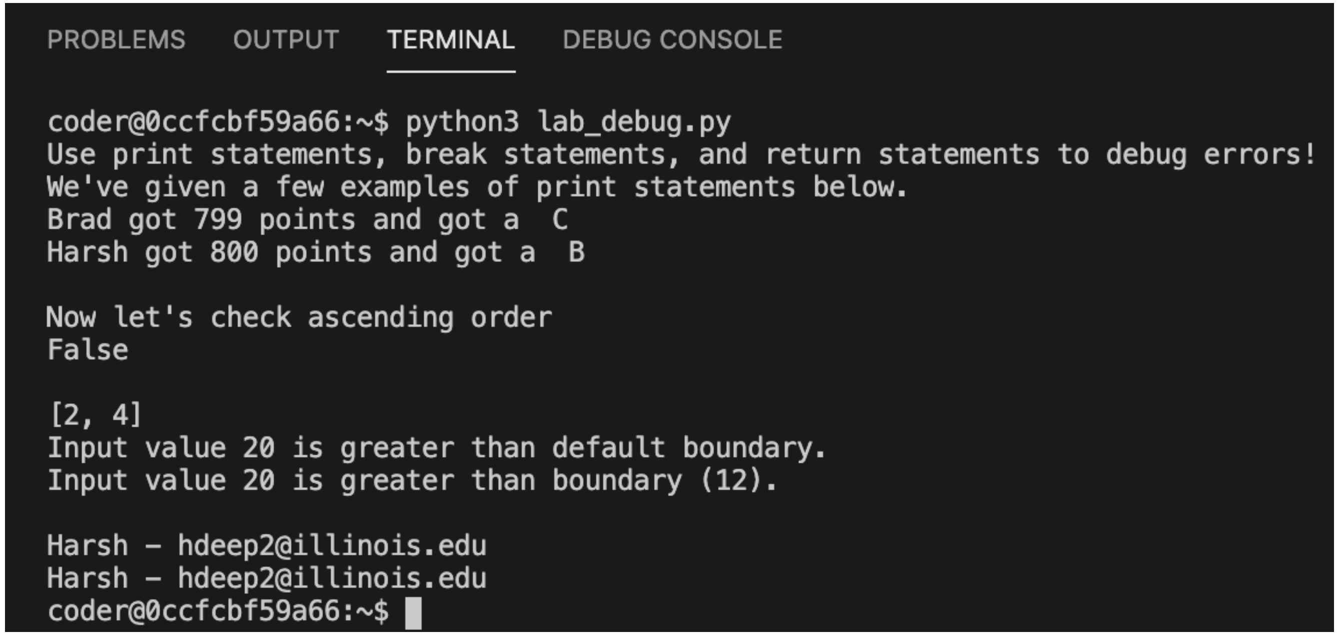
Practice identifying and correcting errors in code

Review function overloading and class definitions

Review the vscode workspace format in PrairieLearn

Running Python in vscode





```
coder@0ccfcbf59a66:~$ python3 lab_debug.py
Use print statements, break statements, and return statements to debug errors!
We've given a few examples of print statements below.
Brad got 799 points and got a  C
Harsh got 800 points and got a  B

Now let's check ascending order
False

[2, 4]
Input value 20 is greater than default boundary.
Input value 20 is greater than boundary (12).

Harsh - hdeep2@illinois.edu
Harsh - hdeep2@illinois.edu
coder@0ccfcbf59a66:~$
```

Debugging Python Code

```
coder@87fcd0ee3699:~$ python3 lab_debug.py
Use print statements, break statements, and return statements to debug errors!
We've given a few examples of print statements below.
Brad got 799 points and got a  D
Harsh got 800 points and got a  D

Our first list is not in sorted order and should return False:
False
Our second list is in sorted order and should return True:
False

What is the correct answer for removeOdds here? Is your solution doing it?
Traceback (most recent call last):
  File "lab_debug.py", line 117, in <module>
    print(removeOdds(l1))
  File "lab_debug.py", line 46, in removeOdds
    val = list_1d[i]
IndexError: list index out of range
```

Read the error message!

The error message will usually tell you **where** and **how** things went wrong

NameError: Problem with a variable

TypeError: Problem with a variable’s type

IndentationError: Problem with whitespace in code

AttributeError: Object doesn’t have a variable or function being called

IndexError: You are trying to access something that doesn’t exist

When in doubt — Google is your friend!

You encounter an error…

```
Traceback (most recent call last):
  File “examples.py”, line 2, in <module>
    x += 5
NameError: name 'x' is not defined
```

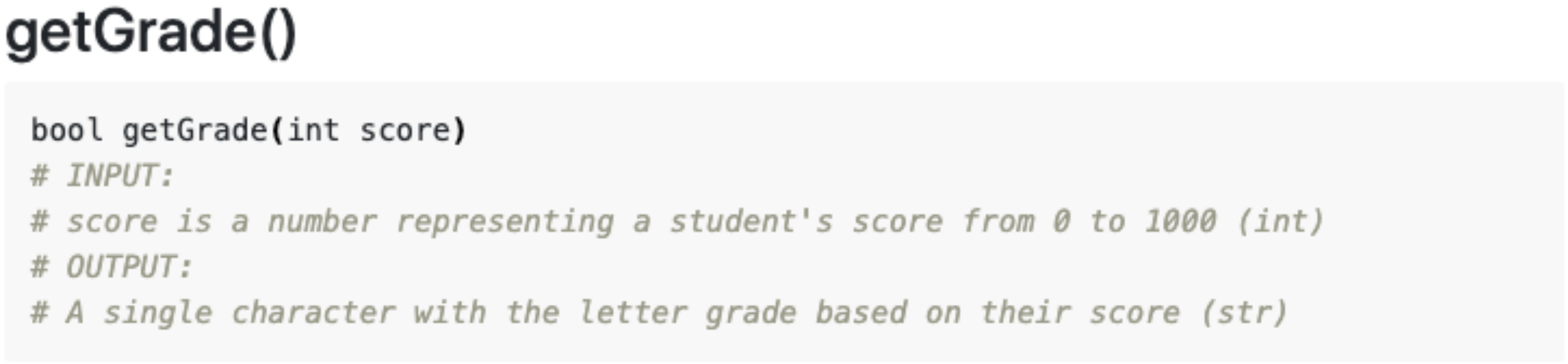
```
Traceback (most recent call last):
  File "examples.py", line 6, in <module>
    print(test[2])
IndexError: list index out of range
```

```
 1
 2 x += 5
 3
 4 test = [0, 3]
 5
 6 print(test[2])
 7
 8
 9
10
11
12
13
```

Not every error has an error message!

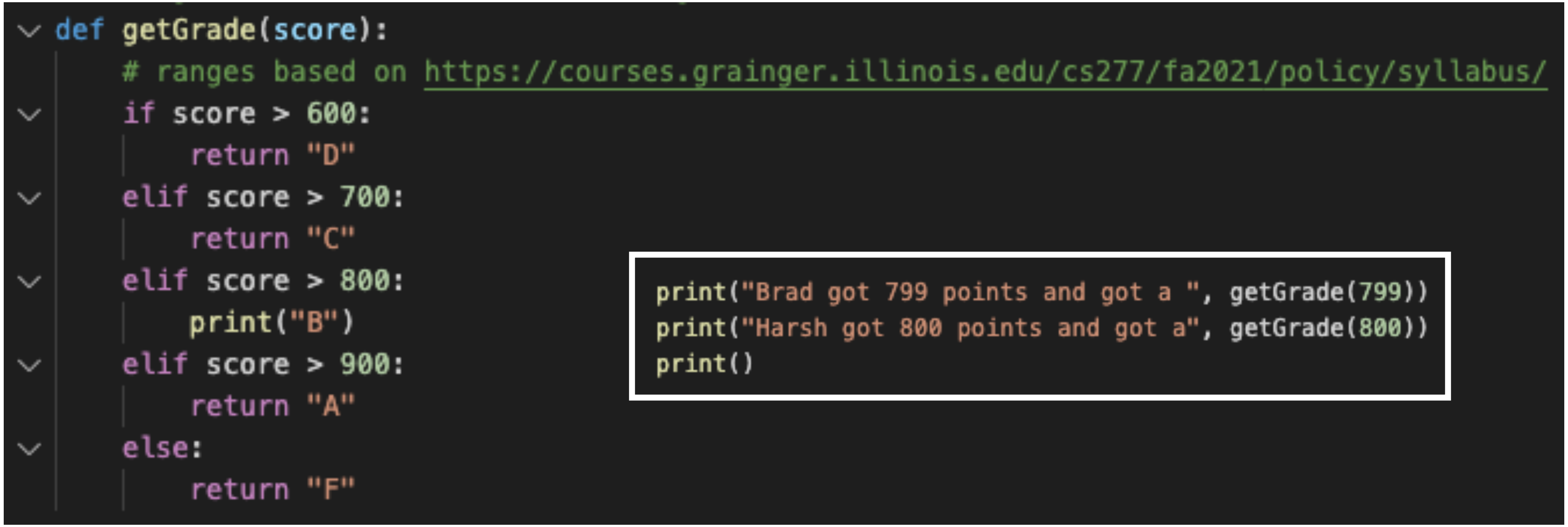
```python
def getReverseEvens(n):
    outList = []
    for i in range(n, 0, -2):
        outList.append(i)

n = 4
ans = [4, 2, 0]

if (getReverseEvens(n)==ans):
    print("Correct!")
else:
    print("Incorrect!")
```

1. Understand the System



getGrade()

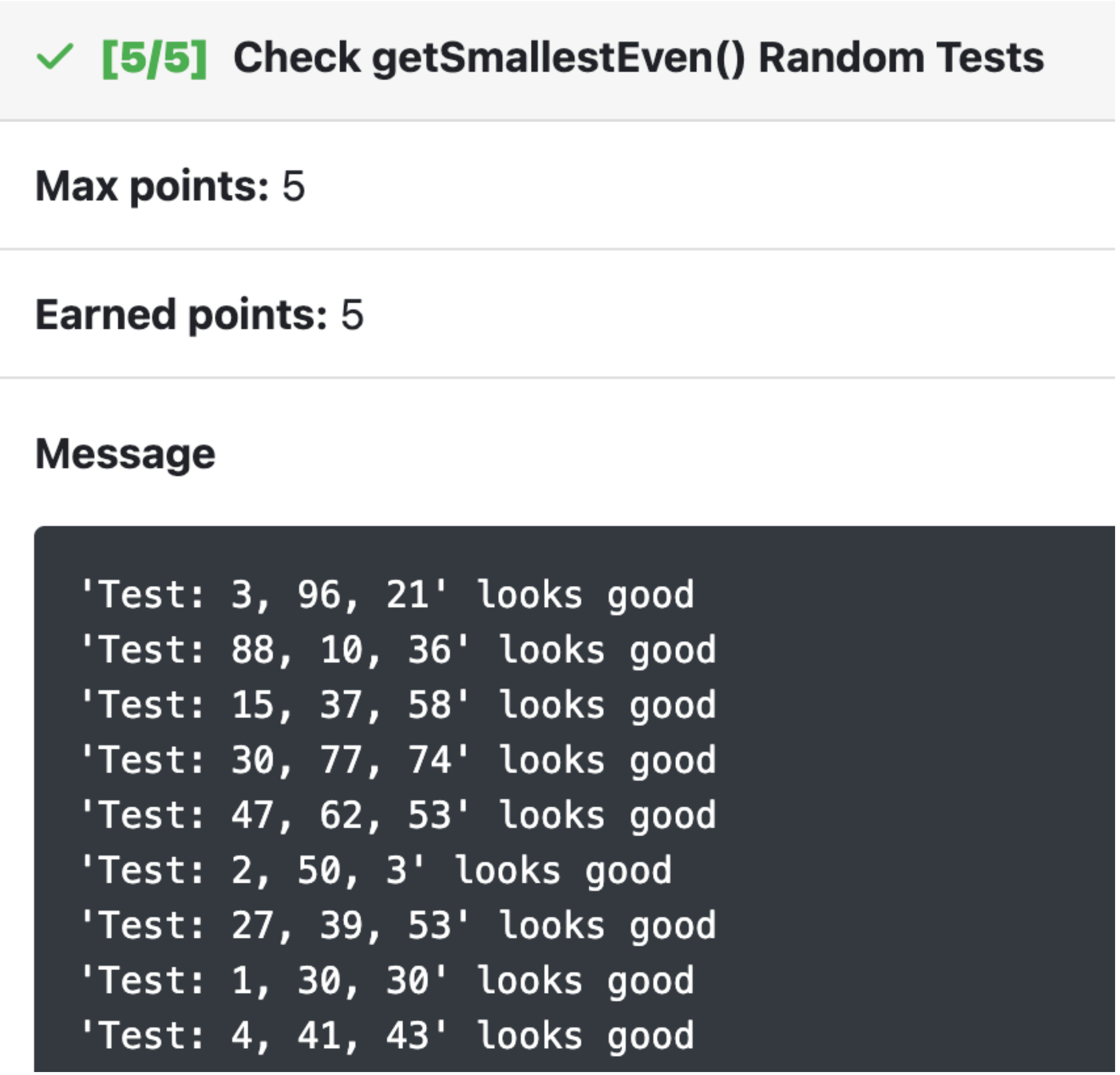
```
bool getGrade(int score)
# INPUT:
# score is a number representing a student's score from 0 to 1000 (int)
# OUTPUT:
# A single character with the letter grade based on their score (str)
```



```python
def getGrade(score):
    # ranges based on https://courses.grainger.illinois.edu/cs277/fa2021/policy/syllabus/
    if score > 600:
        return "D"
    elif score > 700:
        return "C"
    elif score > 800:
        print("B")
    elif score > 900:
        return "A"
    else:
        return "F"
```

```python
print("Brad got 799 points and got a ", getGrade(799))
print("Harsh got 800 points and got a", getGrade(800))
print()
```

2. Make it Fail



✓ [5/5] Check getSmallestEven() Random Tests

Max points: 5

Earned points: 5

Message

```
'Test: 3, 96, 21' looks good
'Test: 88, 10, 36' looks good
'Test: 15, 37, 58' looks good
'Test: 30, 77, 74' looks good
'Test: 47, 62, 53' looks good
'Test: 2, 50, 3' looks good
'Test: 27, 39, 53' looks good
'Test: 1, 30, 30' looks good
'Test: 4, 41, 43' looks good
```

Use the input that failed!

Why did it fail?

Where did it fail?

Correct it and test again!

Does the input work now?

3. Quit Thinking and Look

```python
def getReverseEvens(n):
    outList = []
    for i in range(n, -1, -2):
        outList.append(i)

n = 4
ans = [4, 2, 0]

if (getReverseEvens(n)==ans):
    print("Correct!")
else:
    print("Incorrect!")
```

`print()`

`break / return`

4. Divide and Conquer

```
print("Crash on startup?")

...

print("Crashed yet?")

...

print("Before big function call")

...

print("After big function call")

...

print("Plotting the output")
```

Identify where the problem is

```
Crash on startup?
Crashed yet?
Before big function call
After big function call
```

5. Change One Thing at a Time

```
def getReverseEvens(n):
    outList = []
    for i in range(n, -1, -2):
        outList.append(i)
    return outList

n = 4
ans = [4, 2, 0]

if (getReverseEvens(n)==ans):
    print("Correct!")
else:
    print("Incorrect!")
```

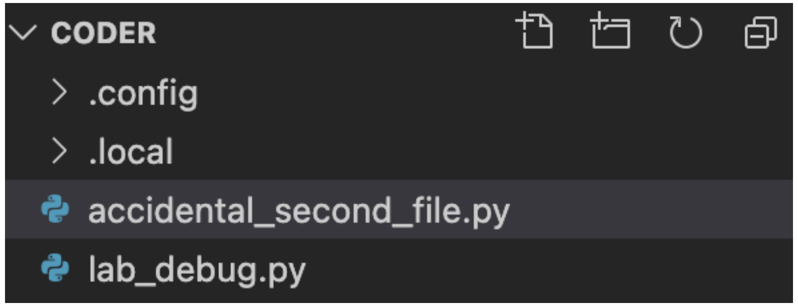
Both a coding and debugging tip!

6. Keep an Audit Trail

Write down the details — great for office hours!

Git commit early and often (with messages!)

7. Check the Plug



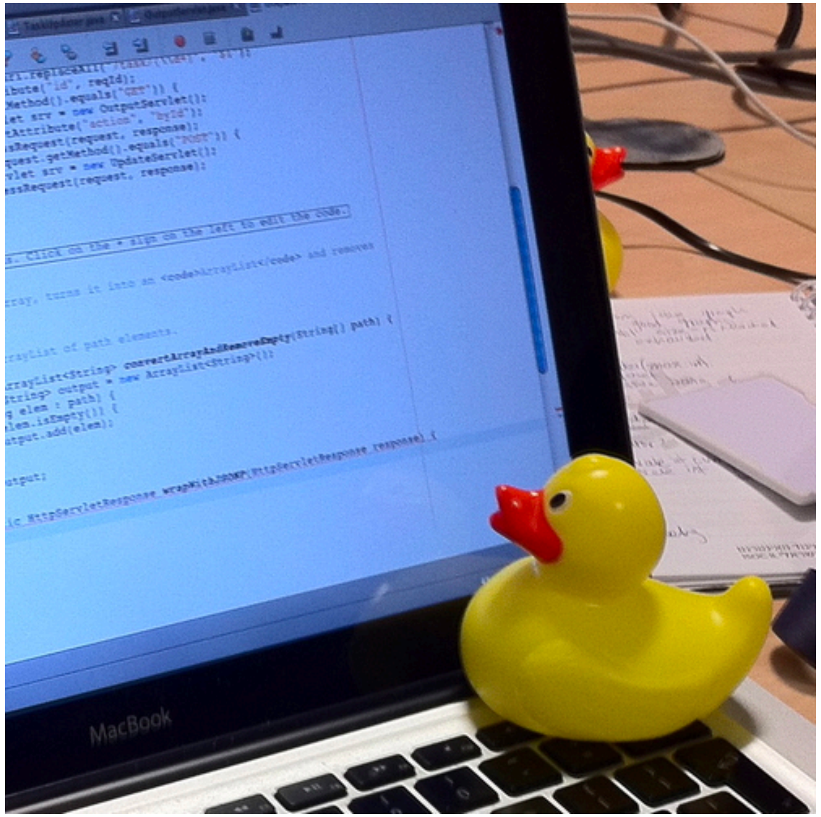
CODER

> .config

> .local

accidental_second_file.py

lab_debug.py



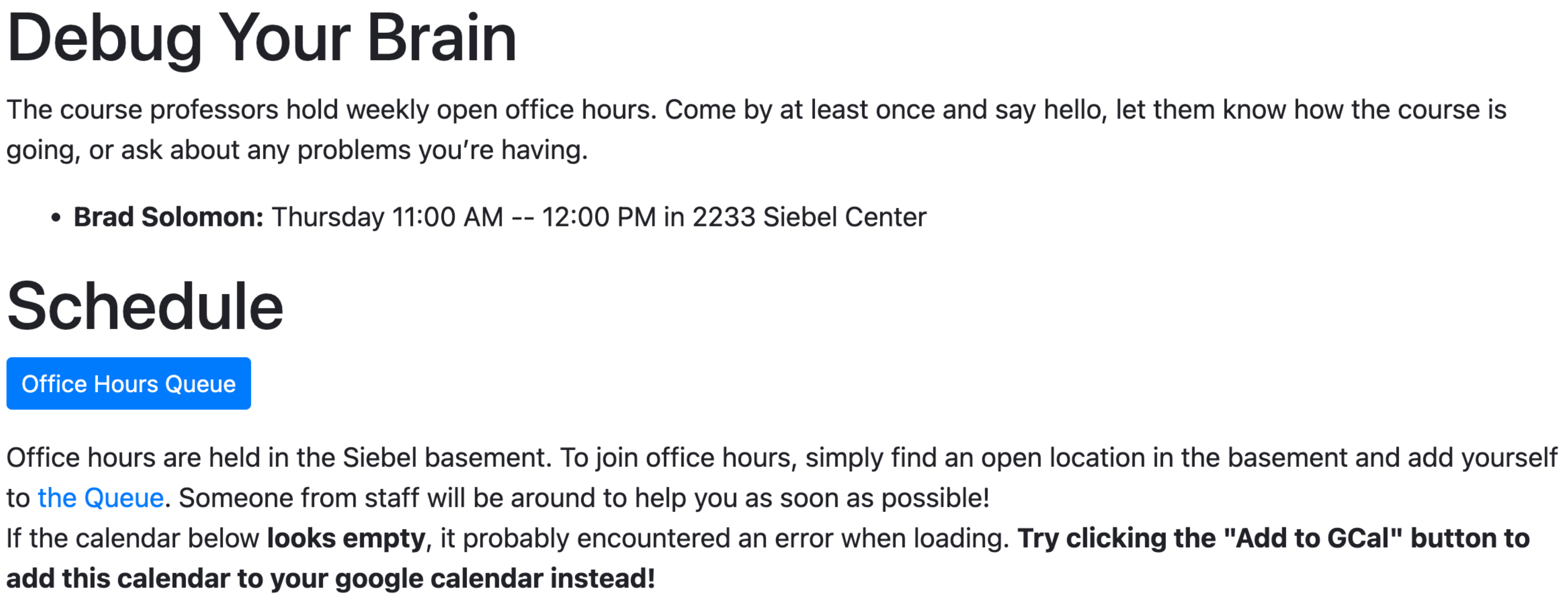
Question your assumptions

Check obvious things!

‘Rubber Duck’ debugging

Is the test wrong?

8. Get a Fresh View



Debug Your Brain

The course professors hold weekly open office hours. Come by at least once and say hello, let them know how the course is going, or ask about any problems you're having.

- **Brad Solomon:** Thursday 11:00 AM -- 12:00 PM in 2233 Siebel Center

Schedule

Office hours are held in the Siebel basement. To join office hours, simply find an open location in the basement and add yourself to the Queue. Someone from staff will be around to help you as soon as possible!
If the calendar below **looks empty**, it probably encountered an error when loading. **Try clicking the "Add to GCal" button to add this calendar to your google calendar instead!**

https://piazza.com/illinois/spring2023/cs277/home

Coding the lab

1) Treat each function as its own independent problem

2) Identify what the function should be doing

3) Correct any errors that are preventing the code from running

4) Correct any errors where the function output is wrong

5) Be aware of edge cases and test your solution thoroughly!