Compsci 101
Lists, Mutation, Objects
Live Lecture

Susan Rodger
Nicki Washington
February 9, 2021

Debugging Steps

Write down what is happening
Brainstorm
Go through list

No

Found problem?

Yes

Fix it!

Announcements

• Assign 1 Totem, due Thursday, Feb 11
• Lab 3 Friday, Do Prelab 3 before lab
  • Note do prework for Feb 11, before Prelab 3
• Sakai QZ due by lecture time each day

• Exam 1 – Tuesday, Feb 16
• Need SDAO letters for exams!
  • Email them to Ms. Velasco
  yvelasco@cs.duke.edu

Genesis Bond ‘16

• Struggled at Duke
  • 5 years
  • Dismissed 1 semester due to grades
• Revature
  • Trainer Full Stack Development
  • She worked smarter
• Facebook Engineer, big success!

“Poor preparation promotes poor performance. In anything you do, your preparation will show.”

Top 10 list - Surviving in CompSci 101

10. Read the book
9. Check Piazza every day
8. Ask Questions
7. Visit office hours/consulting hours
6. Understand what you turn in
Top 10 list (cont)

5. Learn how to debug your programs
4. Think smarter, not harder
3. Follow the 7-step process
2. Seek help (One Hour Rule!)
1. Start programming assignments early

One Hour Rule for Getting Help

- Work on Material
- Stuck
- Has it been an hour?
  - No
  - Get Help
  - Yes

PFTD

- Slicing
- Totem
- Debugging
- List concatenation and nesting
- Mutability
- Objects and what that means
- Exam 1

Exam 1 – Feb 16, 2021

- All topics through Thur. Feb 11 except loops
  - Understand/Study
    - Reading, lectures
    - Assignment 1, APT-1,
    - Labs 0-3 (except for loops in Lab 3)
  - Old tests and solutions on resources tab
    - See recommended ones posted today
- Logistics:
  - Online, More details next time
  - Pick a time to take it on Feb 16
Exam 1 – Feb 16, 2021 (cont)

• What you should be able to do
  • Read/trace code
  • Determine output of code segment
  • Write syntax
• Similar format to Test 1 Fall 2020
  • But note that test covers more topics
  • See posted list of problems posted on calendar page on today’s date

Slicing Python Sequences

• s="hello world"
• l=["my", "big", "beautiful", "world"]
• Slicing provides sub-sequence (string or list)
  • seq[n:m] – all elements i, s.t. n <= i < m
  • Compare s[0:3] and l[0:3]
  • What is length of subsequence? seq[2:4]
  • Compare s[4:-1] and l[2:-1]
  • Is last index part of subsequence?
• We can omit value, e.g., s[2:] or s[:3], good shortcut!

WOTO-1 Slicing

Debugging

• Finding what is wrong + fixing it
  • Finding is its own skill set, and many find difficult
  • Fixing: revisit Step 1—5
How Not To Debug

- Bad (but tempting) way to debug
  - Change a thing. Does it work now?
  - No … another change … how about this?
- Trust doctor if they say?
  - “Ok try this medicine and see what happens?”
- Trust mechanic if they say?
  - “Let’s replace this thing and see what happens”

It may be easy, but that doesn’t make it a good idea!

Debugging Steps

1. Write down exactly what is happening
   1. input, output, what should be output
   2. ____ happened, but ____ should happen
2. Brainstorm possible reasons this is happening
   1. Write down list of ideas
3. Go through list
4. Found it?
   1. Yes, fix it using the 7-steps
   2. No, go back to step 2

Remember: One-hour rule

Debugging Steps

- Write down what is happening
- Brainstorm
- Go through list
- Found problem?
  - No
  - Yes!
- Fix it!

WOTO-2 – Relate W’s to Debugging


- Who was involved?
- What happened?
- Where did it take place?
- When did it take place?
- Why/How did it happen?

Translate these questions to debugging
WOTO-2 – Relate W’s to Debugging

- Who was involved?
  - Which variables are involved?
- What happened?
  - What kind of error/bug is it?
- Where did it take place?
  - Where in the code did this happen?
- When did it take place?
  - Does it happen every time? For certain cases?
- Why/How did it happen?
  - Given the answers to the above, how did the error/bug happen?

Step 7 -> Steps 1-4 or 5

Which year is a leap year?

- A Leap Year must be divisible by four.
- But Leap Years don't happen every four years … there is an exception.
  - If the year is also divisible by 100, it is not a Leap Year unless it is also divisible by 400.

WOTO: Buggy Leap Year

```python
def is_leap_year(year):
    if year % 4 == 0:
        return True
    if year % 100 == 0:
        return False
    if year % 400 == 0:
        return True
    return False
```

Input: 1900
Output: True
Should be: False
WOTO: Buggy Leap Year

Who? (Which variables)

- year

What kind of bug is it?

- Semantic error

Where in the code?

- One of the places it returns True

When does it happen?

- Input: 1900, but not 2016 nor 2019

Why/How did it happen?

- A property 1900 has but not 2016 and 2019

---

Another Example:
Function withCutOff

- This function should calculate an overall quiz score, using the total points of all your quizzes.
- If you earn 75% or more of the total points you get a 100% or 1.0
- If you earn less than 75% then your score is the total number of points you have, divided by the number of points that would represent 75% of the score.

---

Why Leap Year Buggy?

- Why: Should not always return True if year is divisible by 4
- Solution: Check first for %400, then %100, and finally %4

---

Input: 1900
Output: True
Should be: False
withCutOff Function Examples

- **Example 1**, total points is 100, you have 90 points
  - 75% of points is 75 points, you have many more
  - Your score is 100% or 1.0.
- **Example 2**, total points is 100, you have 60 points
  - 75% of points is 75
  - your score is 60/75 is 80% or 0.8.
- **Example 3**, total points is 134, you have 50 points
  - 75% of points is 100, (134*0.75 is 100)
  - Your score is 50/100 is 50% or 0.5.

WOTO: Buggy withCutOff function

```python
7    def withCutOff(total, possible):
8        denominator = int(possible*0.75)
9        percent = total/denominator
10       if percent > 1:
11           percent = 1.0
12       return percent
```

Input: (1,1)
Output: Error
Should be: 1.0

Why is it 0? Where does it get its value?

WOTO: Buggy withCutOff function

- Who? (Which variables)
  - total, denominator
- What kind of bug is it?
  - Runtime error
- Where in the code?
  - Line 9
- When does it happen?
  - Input (1,1), but not (75,100) nor (50,134)
- Why/How did it happen?
  - Divide by zero, so denominator variable is zero

Input: (1,1)
Output: Error
Should be: 1.0
Why Score Buggy?

• Why: Not accounting for possibility of rounding down to 0
• Solution: Check if denominator is 0 and have special case

Mutating Lists

• \(lt = ['Hello', 'world']\)
  • Change to: ['Hello', 'Ashley']
• Concatenation: \(lt = [lt[0]] + ['Ashley']\)
• Index: \(lt[1] = 'Ashley'\)

• How change ‘b’ in \(lt = [1, 'a', [2, 'b']]\)?
  • \(lt[2][1] = 'c'\)

WOTO-5 List Mutation