On your way in...(on the side table)

Pick-up:
1. Homework 01 print-out
2. POGIL Activity #12
3. POGIL Activity #13
4. Day of the Week Algorithm print-out
Welcome to CS 134!

Introduction to Computer Science

Iris Howley

-Functions-

Spring 2019
Clare Boothe Luce Scholars App Due Friday

• If you are a:
  • Sophomore woman
  • Planning to major in Computer Science, Astrophysics, Geosciences, Math/Stat, or Physics
  • A US Citizen (required by the funding agency)
  • Interested in research

• CBL is a really nice funding opportunity for you to do research, travel to conference, purchase research supplies, etc.

• Applications due this Friday, February 8

http://science.williams.edu/student-faculty-research/clare-boothe-luce-scholars/
Homework 01

• Due Monday, February 11 (in less than a week), in class

• Some open-ended responses to get you to think about why we do some of the things we do, in programming

• A little bit of code reading

• A little bit of code writing

• A little bit of getting to know your TAs
Labs are due Thursday and Friday (at noon)

- If you have Monday lab:
  - git push your work by Thursday at noon!

- If you have Tuesday lab:
  - git push your work by Friday at noon!

For every lab!
(unless stated otherwise)
Note:
Homeworks that you turn in are marked as “Homework”

POGIL activities are in-class, optional activities that are not turned in.
(but they’re meant to assist in your learning)
(if you are struggling with concepts in the POGIL activity, you’ll encounter the same struggles in other parts of this course)
Computer Science 134C – Spring 2019
Iris Howley & Duane A. Bailey
Homework 1 – Due: Monday, in class

Please turn in answers to the following questions on Monday, in class.

Name: _______________________________ Partner: _______________________________

Python Activity 12: Void Functions

Learning Objectives
Students will be able to:
Content:
<table>
<thead>
<tr>
<th>Week of</th>
<th>Monday</th>
<th>LAB</th>
<th>Wednesday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb. 1</td>
<td>—</td>
<td>I. PYTHON AND GIT</td>
<td>3. Functions (TP3)</td>
<td>1. Hello, world! (TP1)</td>
</tr>
<tr>
<td>Feb. 4</td>
<td>2. Expressions (TP2)</td>
<td>II. PROCEDURE</td>
<td>6. Iteration (TP7)</td>
<td>4. Conditions (TP5-6)</td>
</tr>
<tr>
<td>Feb. 11</td>
<td>5. Abstraction (TP4)</td>
<td>III. TOOLBOX BUILDING</td>
<td>8. Interpretation</td>
<td>Winter Carnival</td>
</tr>
<tr>
<td>Feb. 18</td>
<td>7. Strings (TP8-9)</td>
<td>IV. DEBUGGING</td>
<td>11. Files (TP14)</td>
<td>9. Lists, Tuples (TP10,12)</td>
</tr>
<tr>
<td>Mar. 11</td>
<td><em>Slack</em></td>
<td><em>Spring Break</em></td>
<td>19. Linked Lists</td>
<td>17. Classes</td>
</tr>
<tr>
<td>M. 18&amp;25</td>
<td><em>Spring Break</em></td>
<td>VII. CLASS DESIGN</td>
<td>22. <em>Slack</em></td>
<td><em>Spring Break</em></td>
</tr>
<tr>
<td>Apr. 22</td>
<td>27. Images</td>
<td>X. JAVA</td>
<td>34. <em>Slack</em></td>
<td>29. Object Persistence</td>
</tr>
<tr>
<td>Apr. 29</td>
<td>30. Java I.</td>
<td>X. JAVA (CONT.)</td>
<td></td>
<td>32. Java III.</td>
</tr>
<tr>
<td>May 6</td>
<td>33. Java IV.</td>
<td></td>
<td></td>
<td>35. Evaluations</td>
</tr>
</tbody>
</table>
Have you been following along in the textbook?

---

### Lab 1
- **Date:** February 4
- **Topic:** Introduction to the Python/git workflow

### Homeworks
- **Due Date:** February 4 (in-class)
- **Topic:** Homework 0. Data and algorithms

### Resources
- **The Textbook**
- **Typical workflows**
- **Duane's Incredibly Brief Intro to Unix and Emacs**
A Thought.
IT IS OKAY TO MAKE MISTAKES.
THIS IS HOW WE LEARN.

IT IS OKAY FOR ME TO MAKE MISTAKES.
I WILL MAKE A LOT OF MISTAKES.

The longer the program, the more errors!
Even for experts!
YOU ARE MY PAIR PROGRAMMING PARTNERS.
Admiral Grace Murrany Hopper

Harvard Mark II

9/9

0:400
1000
Anton started
- Anton
1330 1530 MP - MC
033
PROZ 2
2.13097245
Coned
2.13097245
Relays 0-2 = 033 found specialspeed test
in Relay
Relays changed
Started cosine tape (same check)
15:25
Started mult. adder test
15:45
Relay 70 panel - (moth) in relay.
163/100
First actual case of bug being found.
1700
Closed down.
...back to the lesson...
TODAY’S LESSON

Programs are useful because they are reusable.

(among other reasons)
Libraries
Process-Oriented Guided-Inquiry Learning (POGIL)
POGIL

• Look at Python Activity 12
• Find a partner and talk through question 1 together

• When time is up, we’ll execute the code as a class.
Look at POGIL Activity #12

```python
# Function definition
def printMessage():
    print("You're a wizard, Harry")
    print("I’m a wut?")

# Function definition
def main():
    print("Hello Programmer!")
    # Function call
    printMessage()

# Function call
main()
```
Interpreting an Algorithm
Pixel, the Sentient Snowball
2. Compute the sum of the following quantities:

- the month adjustment from the given table (e.g., 6 for Admiral Hopper)
- the day of the month
- the year (since 1900) = 118
- the whole number of times 4 divides the year (e.g., 29 for Pixel)

\[2 + 16 + 118 + 29 = 165\]
3. Compute the remainder of the sum of step 2, when divided by 7. The remainder gives the day of the week, where Saturday is 0, Sunday is 1, etc. Notice that we can compute the remainders before we compute the sum. You may also have to compute the remainder after the sum as well, but if you’re doing this in your head, this considerably simplifies the arithmetic.

165\%7 = 4

Sat. = 0; Sun. = 1; Mon = 2, Tues = 3, Wed = 4

Pixel was born on a Wednesday
DayOfWeek “Lecture 3” Hand-out

• Look at the algorithm on one side

• Can you see where it is represented in the python code on the other side?

(There are some more advanced topics in the python code, like lists & if statements we haven’t yet covered)
QUESTIONS?
Leftover Slides
Grace Hopper, December 9 1906

2. Compute the sum of the following quantities:

- the month adjustment from the given table (e.g., 6 for Admiral Hopper)
- the day of the month
- the year *(since 1900) = 6*
- the whole number of times 4 divides the year (e.g., 29 for Pixel)

1

6+9+6+1 = 22
Grace Hopper, December 9 1906

3. Compute the remainder of the sum of step 2, when divided by 7. The remainder gives the day of the week, where Saturday is 0, Sunday is 1, etc. Notice that we can compute the remainders before we compute the sum. You may also have to compute the remainder after the sum as well, but if you’re doing this in your head, this considerably simplifies the arithmetic.

$$22 \% 7 = 1$$

Saturday = 0
Sunday = 1

Admiral Grace Murray Hopper was born on a Sunday