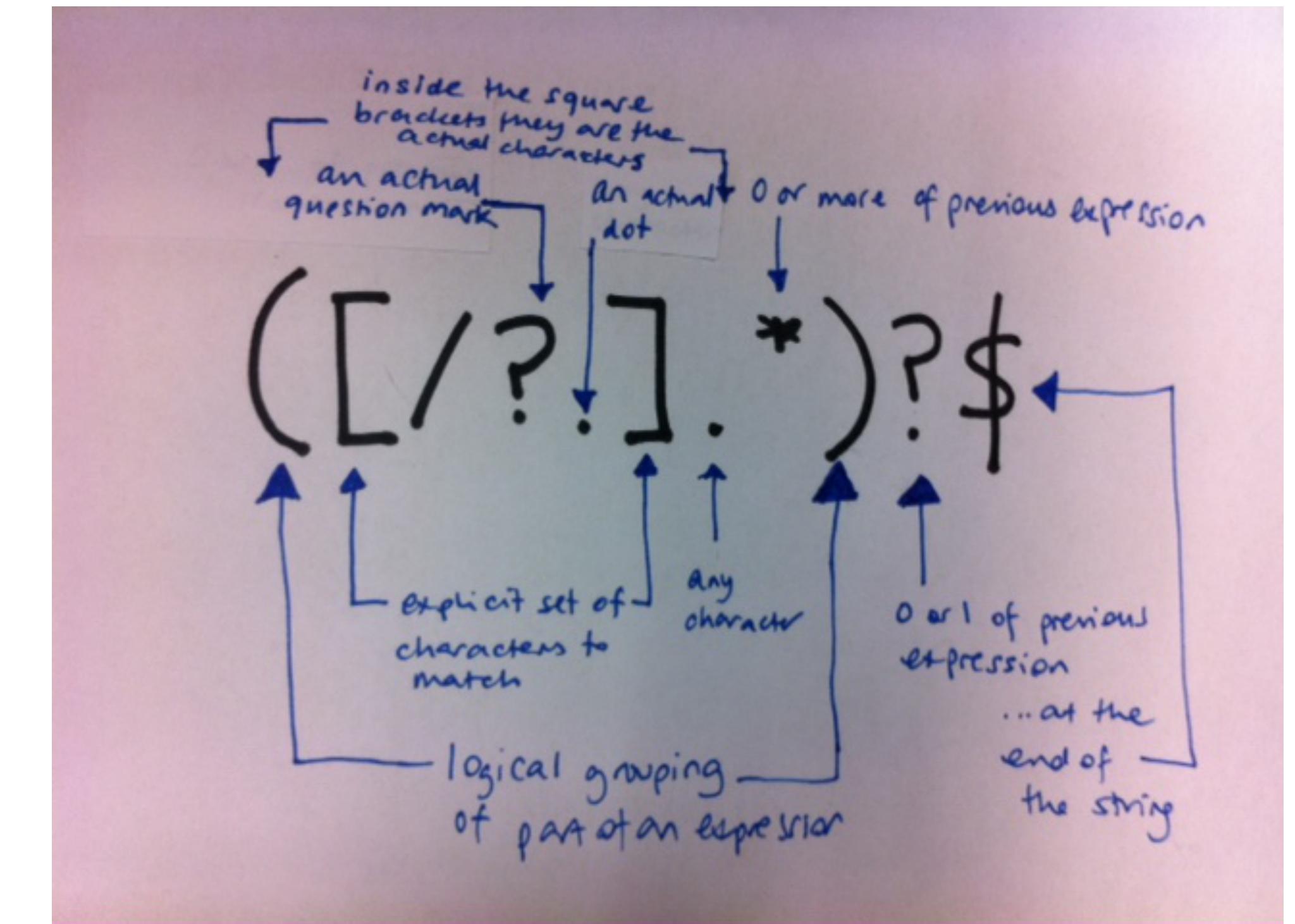
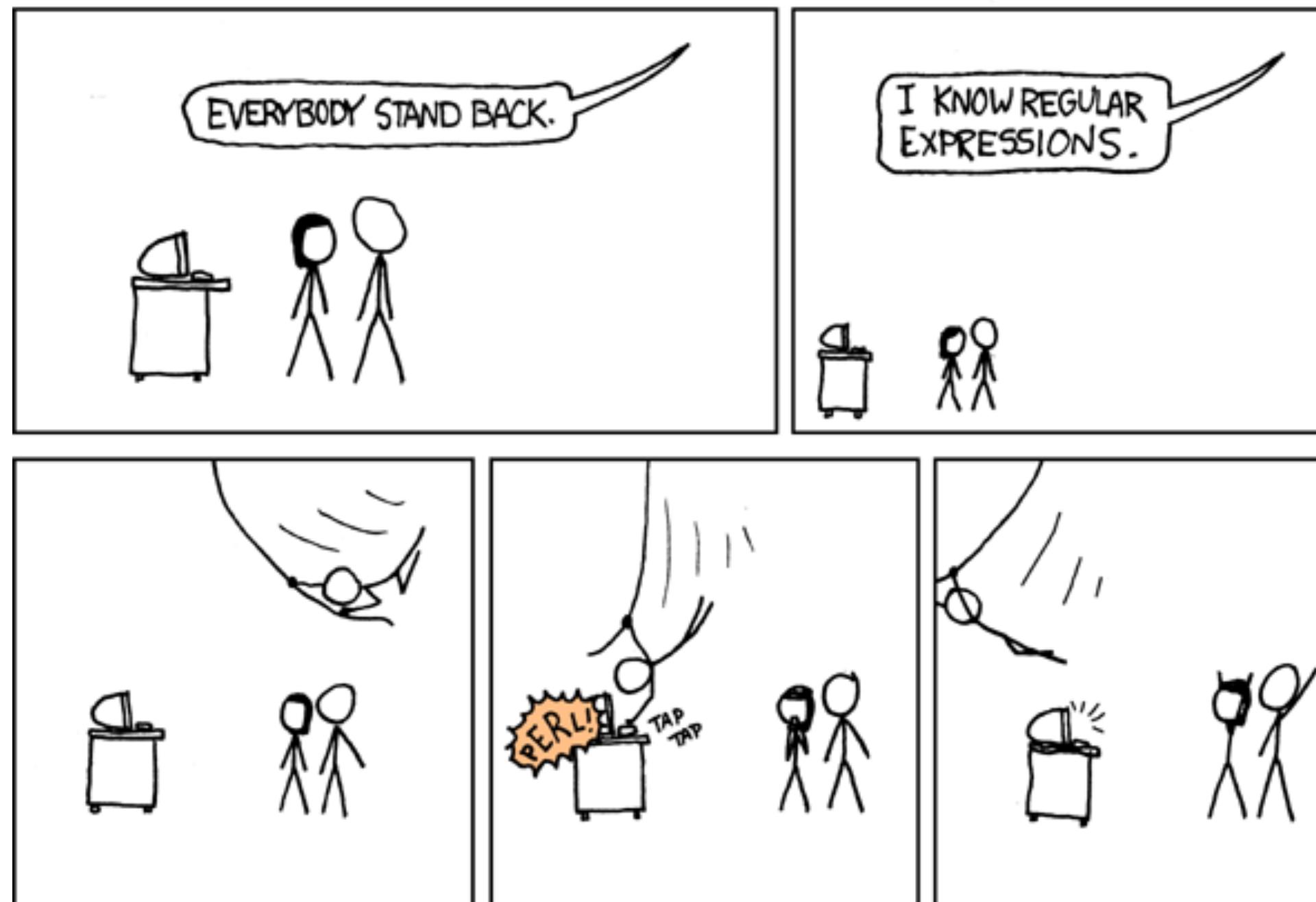


Intro & Regular Expressions

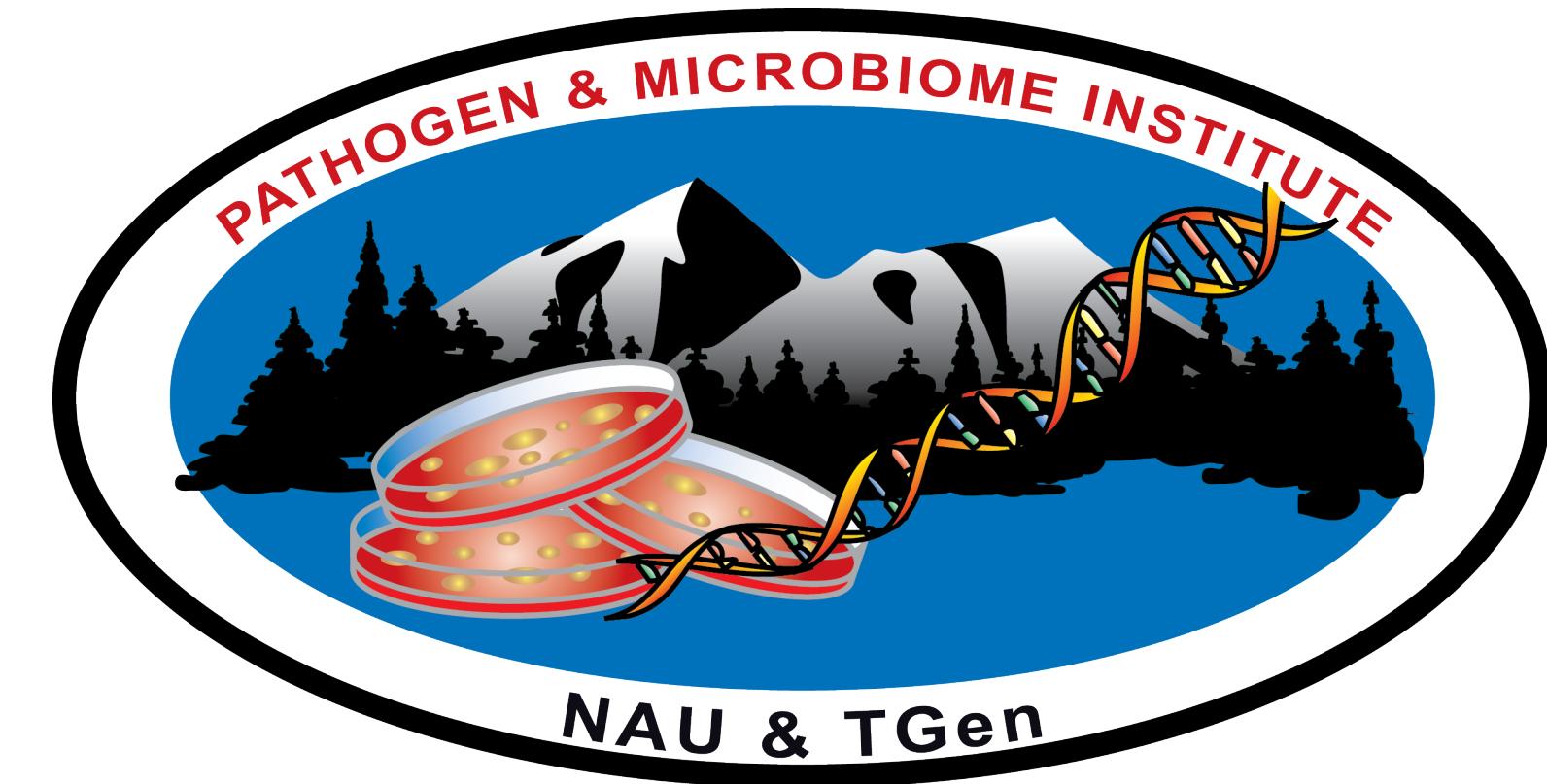
Spring 2021
PCfB Class 1
January 15, 2021



Outline

- Introductions
- Course organization
- Plain text files
- Regular expressions

PhD - Evolutionary genetics



PostDoc - Pathogen genomics



USAMRIID

United States Army
Medical Research Institute
of Infectious Diseases

Biodefense solutions to protect our nation



**Assistant Professor
Dept of Biological Sciences
Pathogen and Microbiome Institute**

Intros

- 1. Your name**
- 2. Your research focus**
- 3. What you hope to get from this class**
- 4. (Optional) Pronouns**

Course
organization

What this course is:

- Intro to general computing techniques broadly applicable to many research-related tasks

What it isn't:

- A bioinformatics class

Most course materials on



https://github.com/jtladner/PracticalComputing_Spring2021

“Pulling” GitHub updates

Syllabus on Bb Learn

The screenshot shows the Bb Learn course interface for 'BIO-599 (1211-4804) CONTEMPORARY DEVELOPMENTS (Spring 2021 M16) 007 Topic - PRACTICAL COMPUTING IN BIOLOGY'. The left sidebar lists course navigation options: Course Content, Syllabus and Schedule, Assignments, Discussions, My Grades, Course Messages, Announcements, Tests/Quizzes, and NAU Help. The main content area displays the 'Syllabus and Schedule' page, featuring the NAU logo and a banner placeholder for 'SYLLABUS & SCHEDULE'. A file icon with a document and a download arrow is shown below the banner.

BIO-599 (1211-4804)

CONTEMPORARY
DEVELOPMENTS (Spring
2021 M16) 007 Topic -
PRACTICAL COMPUTING
IN BIOLOGY

Course Content

Syllabus and Schedule

Assignments

Discussions

My Grades

Course Messages

Announcements

Tests/Quizzes

NAU Help

Syllabus and Schedule

Build Content Assessments Tools Partner Content

banner placeholder

NAU

SYLLABUS & SCHEDULE

Syllabus & Schedule

Assignments submitted via Bb Learn

The screenshot shows a Bb Learn course interface. On the left, a sidebar lists course details: BIO-599 (1211-4804), CONTEMPORARY DEVELOPMENTS (Spring 2021 M16) 007 Topic - PRACTICAL COMPUTING IN BIOLOGY. Below this are links for Course Content, Syllabus and Schedule, Assignments (selected), and Discussions.

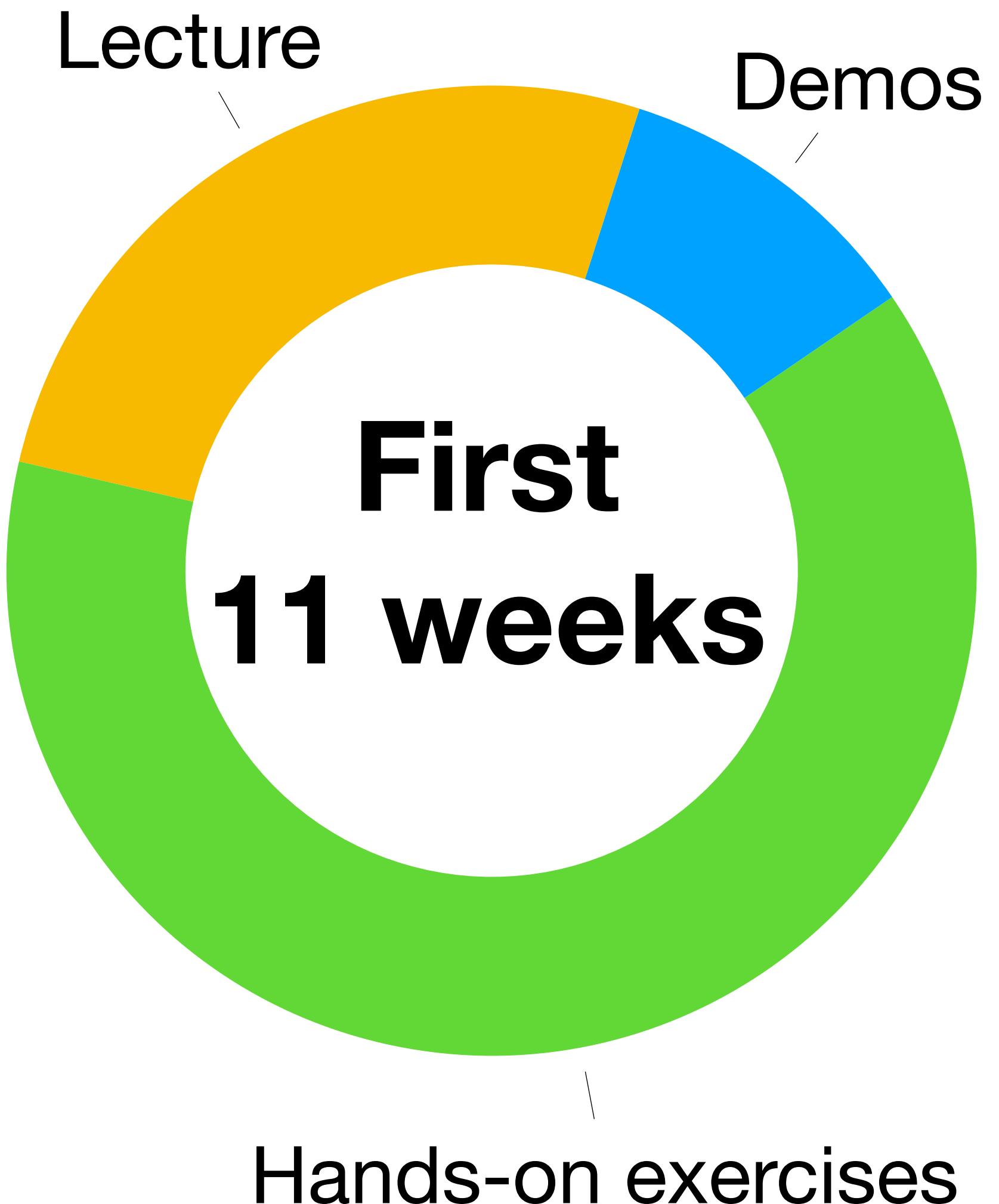
The main area is titled "Assignments". It features a navigation bar with "Build Content", "Assessments", "Tools", and "Partner Content". A specific assignment titled "Week 01 - Regular Expressions" is displayed. It includes a document icon, the assignment title, and a download link for "Assignment_01.docx" (17.936 KB). A descriptive text below states: "Assignment for Week 01 - Regular Expressions."

Required text



- Haddock, S. H. D. and Dunn, C. W. (2010). Practical Computing for Biologists. Sinauer Associates
- <http://practicalcomputing.org/>
- Reading must be complete **PRIOR** to class

Class organization



Individual projects (Last 5 weeks)

- Individual coding projects
- Topic of your choice
- 3-4 work weeks
- 1-2 weeks for presentations

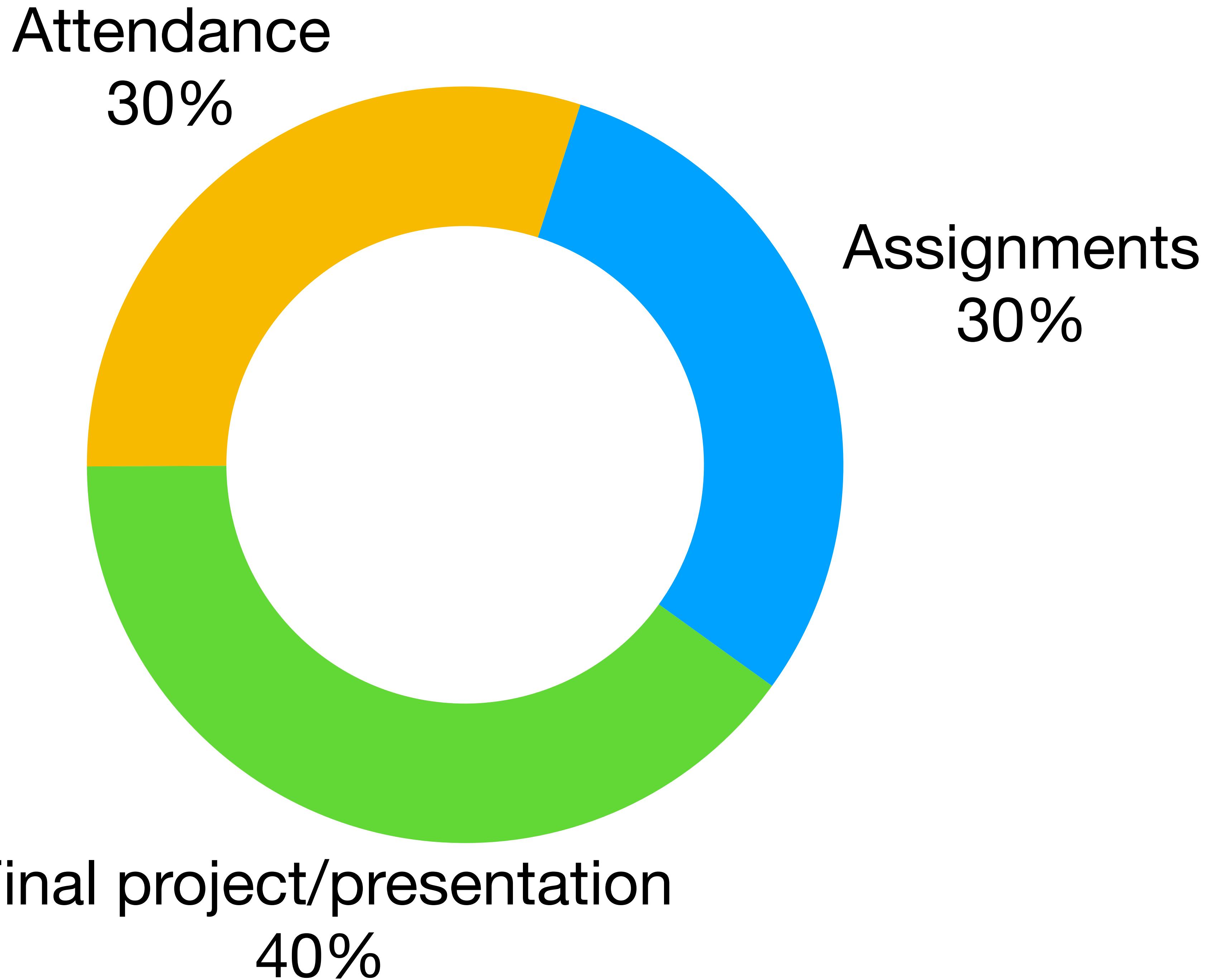
Assignments

- One assignment per week (weeks 1-11)
- Focus on hands-on time in class (may need to complete outside of class)
- Always due by 11:59 pm on Tuesday
- Directions on GitHub, Answer sheet on Bb Learn
- Partial credit for revisions

Breakout groups for hands-on exercises

- Main room: questions/troubleshooting w/ professor
- Quiet workspace: for independent work
- Discussion spaces: for discussion among students

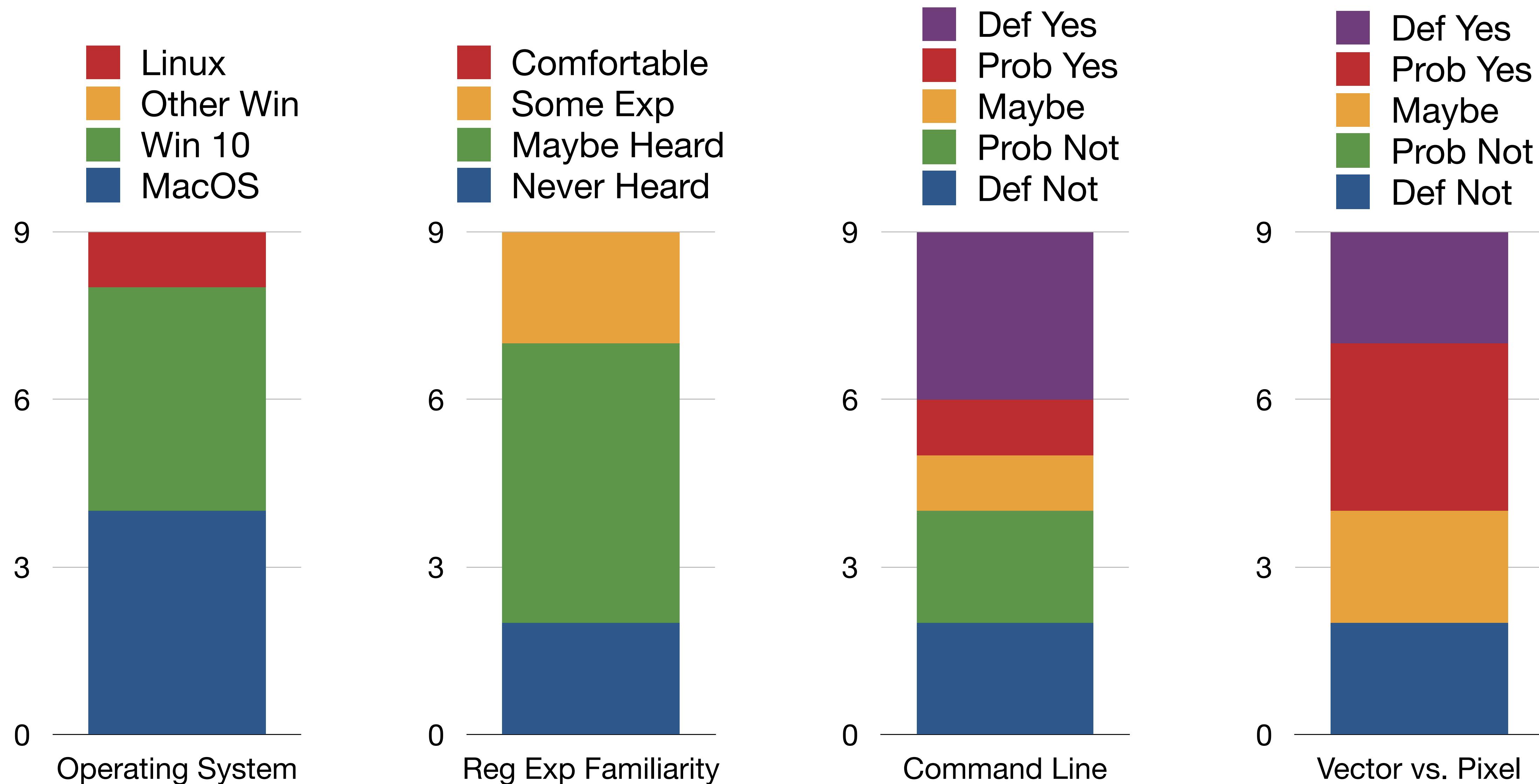
Grading



Final project - deadlines

Week	Date	Topic	Reading
1	1/15	Intro, Setup & Regular Expressions	PCfB: Ch. 1-3
2	1/22	The Shell - Part 1	PCfB: Ch. 4-5
3	1/29	The Shell - Part 2	PCfB: Ch. 6, 21
4	2/5	Python Programming - Part 1	PCfB: Ch. 7-8 Jupyter Tutorial
5	2/12	Python Programming - Part 2	PCfB: Ch. 9
6	2/19	Python Programming - Part 3	PCfB: Ch. 10-11
7	2/26	Python Programming - Part 4	PCfB: Ch. 12
8	3/5	Debugging, Combining Methods	PCfB: Ch. 13-14
9	3/12	Graphical concepts: vectors vs. pixels	PCfB: Ch. 17-19
10	3/19	Making Figures in Python - Part 1	Matplotlib overview
11	3/26	Making Figures in Python - Part 2 (*Project proposal due)	
12	4/2	Work/Troubleshoot Day #1	
13	4/9	Work/Troubleshoot Day #2	
14	4/16	Work/Troubleshoot Day #3	
15	4/23	Project Presentations - Part 1	
16	4/30	Project Presentations - Part 2	
Finals	5/3	*Final project due	

Pre-class survey



Computer setup

- Text Editor
- GitHub Repository
- Command line terminal

[https://github.com/jtladner/PracticalComputing_Spring2021/
tree/master/Getting%20Started](https://github.com/jtladner/PracticalComputing_Spring2021/tree/master/Getting%20Started)

Plain text

files

Plain text file

- Pure sequence of character codes
- No formatting (e.g., text size, color, font, spacing)
- Human and machine readable
- Standardized

Which of these formats are NOT plain text?

Excel (.xlsx)

html

OpenOffice (.odf)

Google Sheet

text (.txt)

fasta

markdown

xml

Google Doc

nexus

json

Word (.doc)

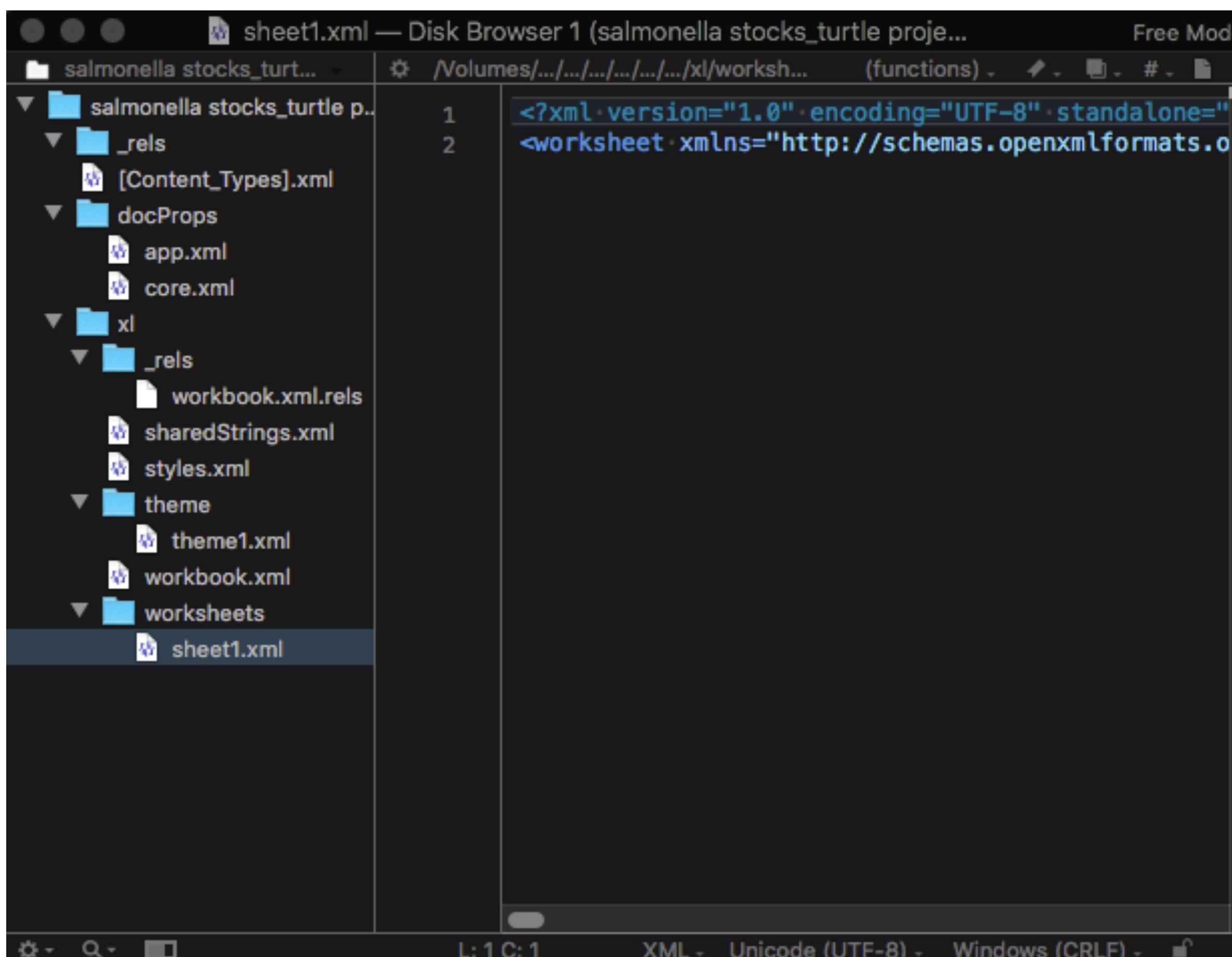
python script (.py)

rich text (.rtf)

phylip

Viewing non-plain text in text editor

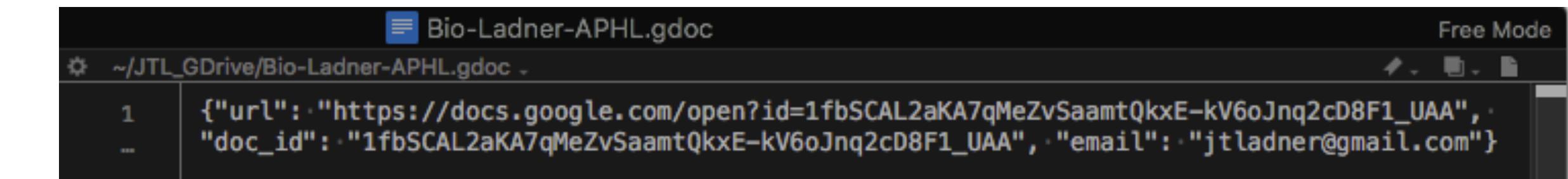
.xlsx/.docx



A screenshot of a file browser window titled "sheet1.xml — Disk Browser 1 (salmonella stocks_turtle proje...)" in "Free Mode". The left pane shows a tree view of an .xlsx file structure under the root folder "salmonella stocks_turtle.pptx". The "worksheets" folder contains "sheet1.xml". The right pane displays the XML content of "sheet1.xml".

```
<?xml version="1.0" encoding="UTF-8" standalone="1">
<worksheet xmlns="http://schemas.openxmlformats.org/spreadsheetml/2006/main">
```

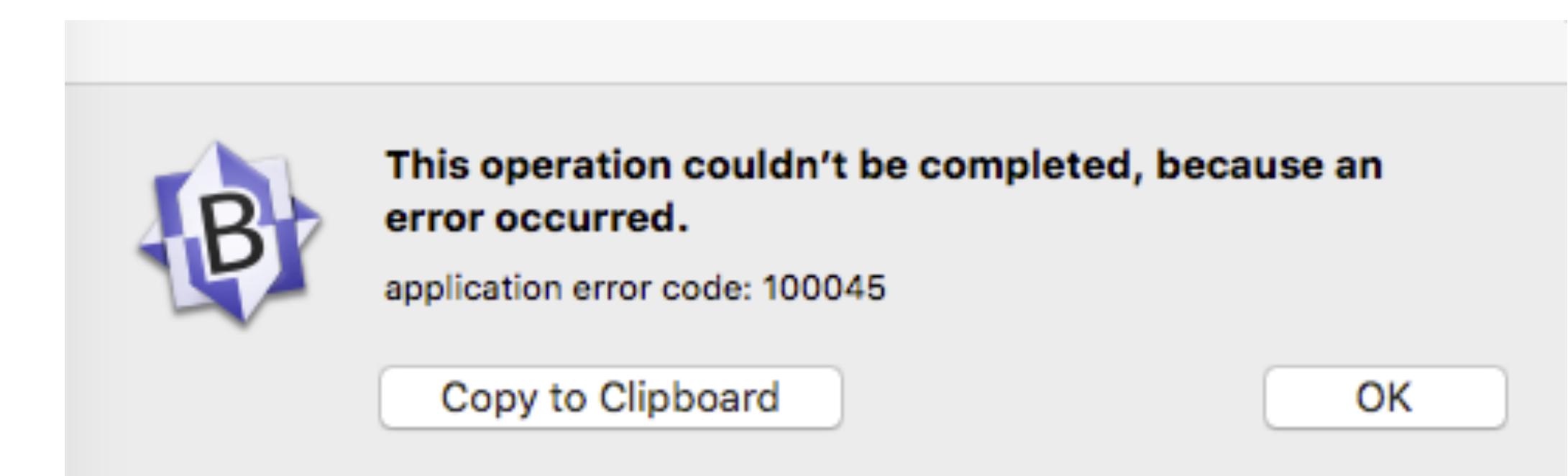
Google Doc



A screenshot of a text editor window titled "Bio-Ladner-APHL.gdoc" in "Free Mode". The content is a JSON object:

```
{"url": "https://docs.google.com/open?id=1fbSCAL2aKA7qMeZvSaamtQkxE-kV6oJnq2cD8F1_UAA", "doc_id": "1fbSCAL2aKA7qMeZvSaamtQkxE-kV6oJnq2cD8F1_UAA", "email": "jtladner@gmail.com"}
```

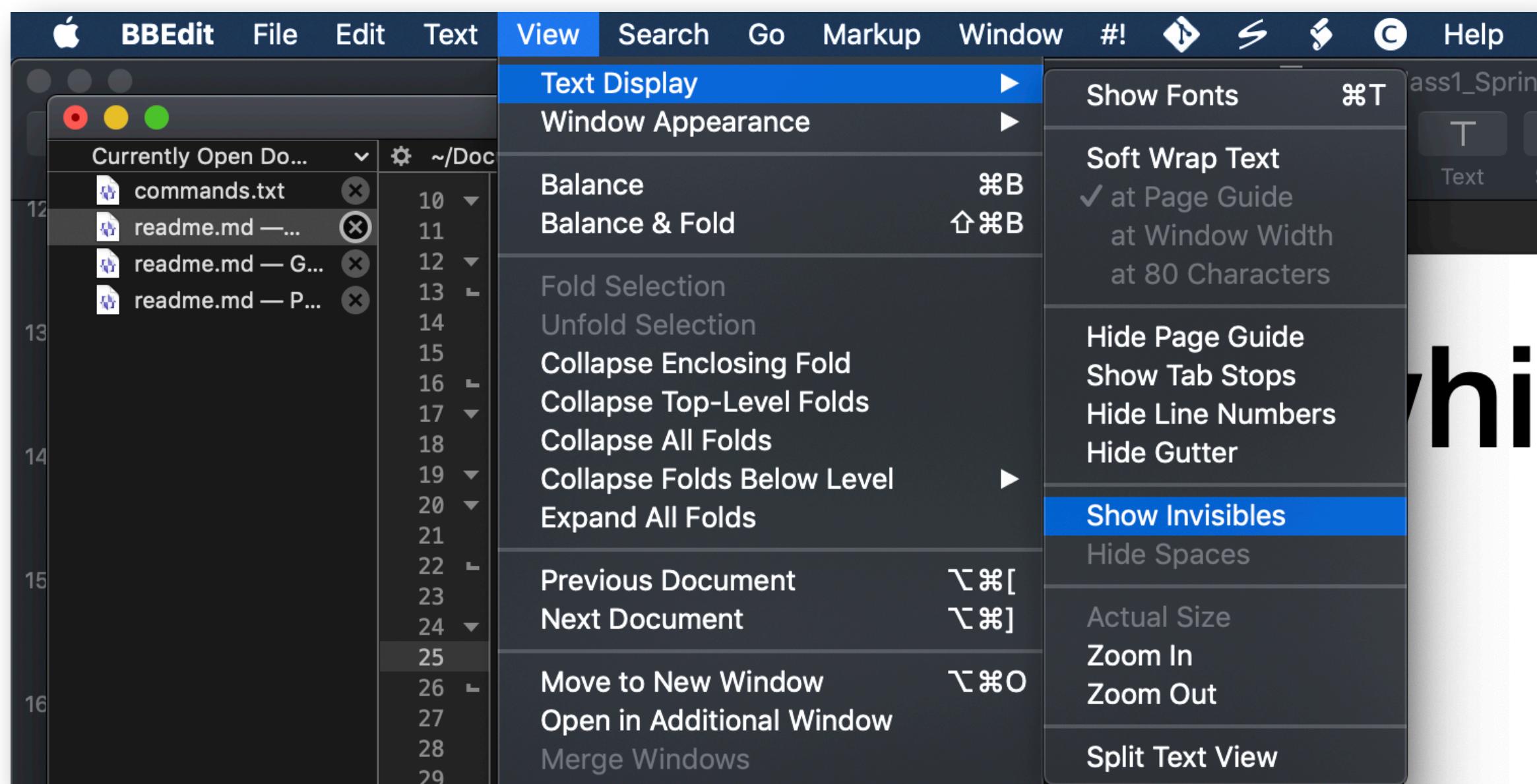
Google Sheet



Whitespace

- Space
- Tab
- End of line

Visualizing white space (BBEdit)



```
10  ### Prep for next class
11
12  1. Open your command line interface and type this command f
13  ``echo $SHELL``
14
15  If the response is not "/bin/bash", let me know.
16
17  ### Assignment
18
19  1. Go to [RegexOne](https://regexone.com/) and complete the
20      - Although the interface will allow you to only match a
21      - Keep track of your solutions in the table provided in
```

```
10  ### Prep for next class
11
12  1. Open your command line interface and type this command
13  ``echo $SHELL``
14
15  If the response is not "/bin/bash", let me know.
16
17  ### Assignment
18
19  1. Go to [RegexOne](https://regexone.com/) and complete
20      - Although the interface will allow you to only matc
21      - Keep track of your solutions in the table provided
```

Visualizing white space (Notepad++)

The screenshot shows the Notepad++ interface with the file "CG-Lion MultiMat Assistant.py" open. The "View" menu is pulled down, and the "Show White Space and TAB" option is selected, indicated by a red border and a checked checkbox. A green circle highlights the main code area, where horizontal white space is represented by yellow lines connecting characters. The code itself is a Python class definition for a Blender operator:

```
class CGL_SetMultiMat(bpy.types.Operator):
    """CG-Lion MultiMat Assistant"""
    bl_idname = "material.setmultimat"
    bl_label = "Set selected objects materials to Principled BSDF"
    def execute(self, context):
        for obj in bpy.context.selected_editable_objects:
            for m in obj.data.materials:
                for n in m.node_tree.nodes:
                    if n.name != 'Material Output':
                        m.node_tree.nodes.remove(n)
                    material_output = m.node_tree.nodes.get('Material Output')
                    grey = m.node_tree.nodes.new('ShaderNodeBsdfPrincipled')
                    grey.inputs['Base Color'].default_value = (0.7, 0.7, 0.7)
```

BBEdit

readme.md

Free Mode

Currently Open Documents

- commands.txt
- readme.md — ...
- readme.md — G...
- readme.md — P...

```
10  ## Prep for next class
11
12  1. Open your command line interface and type this command
13  ``echo $SHELL``
14
15  If the response is not "/bin/bash", let me know.
16
17  ## Assignment
18
19  1. Go to [RegexOne](https://regexone.com/) and complete
20  .... Although the interface will allow you to only match
21  .... Keep track of your solutions in the table provided
22
23  2. We will now start using the text editor on your computer
24  .... 1. Open "EBOV.phy" in your text editor.
25  .... 2. Use a series of find/replace queries to convert the
26  ....
27  .... Fasta format example:
28  .... ``
29  .... >name1
30  .... seq1
31  .... >name2
32  .... seq2
33  .... ``
34
35  .... 3. Edit your regular expression(s) to also change the
36  .... 4. Edit your regular expression(s) to remove all 'N'
37  .... 5. Upload final fasta-formatted file to Bb Learn along
38
39  3. Open "HastingsBirdList\_2007\_.txt" in your text editor
40  .... 1. Design a single search and replace query that uti
```

L: 25 C: 1 Markdown Unicode (UTF-8) Unix (LF)

Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

new 1.txt

```
1 This is a line without any tab or spaces.
2
3 This line contains two spaces but no tab.
4
5 → This line is indicating one tab indent.
6
7 →→ This line is indicating two tab indents.
```

*new 4 - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

new 4

```
1 This is a line of text
2 This is another line of text
3
```

Ln: 3 Col: 1 Sel: 0 | 0 Windows (CR LF) UTF-8 INS

End of line characters differ by OS

- Line feed (LF) - Mac OSX, Linux
- Carriage return (CR) - Mac OS9 and earlier
- Carriage return + line feed (CRLF) - Windows

Regular expressions

Regular expressions

(a.k.a. regex, regexp)

- Powerful find and replace toolkit
- Understood by many text editors, programming languages and even search engines
- Power comes from wildcard operators

\d

\w

\s



\w+

\w*

\w?

[ABC]

[^ABC]

[A-C]

(ABC)

(AB)C

((AB)C)

Λ

\$

Tips

- Try PCfB methodology
 - copy target text into search dialog
 - replace text with wildcards, piece by piece
- Be as specific as possible
- Build in redundancies

Regexp reference tables

Wildcards

\w	Letters, numbers and _
.	Any character except \n \r
\d	Numerical digits
\t	Tab
\r	Return character. Also used as the generic end-of-line character in TextWrangler
\n	Line-feed character. Also used as the generic end-of-line character in Notepad++
\s	Space, tab, or end of line
[A-Z]	A single character of the ranges indicated in square brackets
[^A-Z]	A single character including all characters <i>not</i> in the brackets. Note that this will include \n unless otherwise specified, and may cause you to match across lines
\	Used to escape punctuation characters so they are searched for as themselves, not interpreted as wildcards or special symbols
\\	The \ symbol itself, escaped

Boundaries

^	Match the start of the line, i.e., the position before the first character
\$	Match the last position before the end-of-line character

Quantifiers, used in combination with characters and wildcards

+	Look for the longest possible match of one or more occurrences of the character, wildcard, or bracketed character range immediately preceding. The match will extend as far as it can while still allowing the entire expression to match.
*	As above, matches as many of the previous character to occur, but allows for the character not to occur at all if the match still succeeds
?	Modifies greediness of + or * to match the shortest possible match instead of longest
{}	Specify a range of numbers to repeat the match of the previous character. For example: \d{2,4} matches between 2 and 4 digits in a row [AC]{4,} matches 4 or more of the letter A or C in a row
	Capturing and replacing
()	Capture the search results between the parentheses for use in the replacement term
\1 \$1	Substitute the contents of the matched into the replacement term, in numerical order. Syntax depends on the text editor or language that you are using.

**Questions about the
reading?**



RegexOne

Learn Regular Expressions with simple, interactive exercises.

Exercise 1: Matching Characters

Task	Text	
------	------	--

Match	abcd <ins>efg</ins>	✓
-------	---------------------	---

Match	abcde	✓
-------	-------	---

Match	abc	✗
-------	-----	---

[e-g]+

Continue >

Solve the above task to continue on to the next problem, or read the [Solution](#).

Exercise 1: Matching Characters

Task	Text	
------	------	--

Match	abcdefg	✓
-------	---------	---

Match	abcde	✓
-------	-------	---

Match	abc	✓
-------	-----	---

\w+

Continue >

Solve the above task to continue on to the next problem, or read the [Solution](#).

“Prep for next class”

Class 1 - Jan. 15th 2021

- In this first class we will:
 - Discuss the syllabus and course organization/expectations
 - Troubleshoot computer setup problems
 - Learn to use regular expressions to edit text files

Required Reading (Must be completed ahead of time)

Practical Computing for Biologists, Chapters 1-3

Prep for next class

1. Open your command line interface and type this command followed by 'Enter': `echo $SHELL`

If the response is not "/bin/bash", let me know.

Text editor

regex demos