

Testing, PyCogen scripting and work with Greengenes

Daniel McDonald – Knight Lab

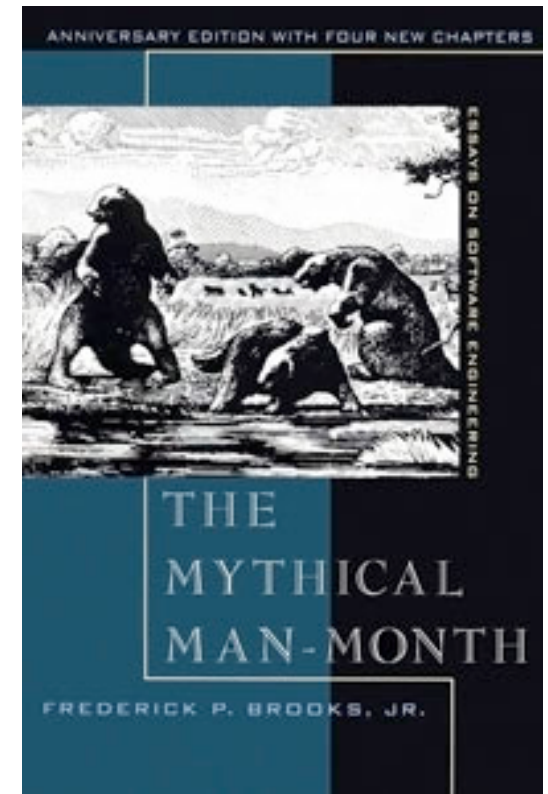
7.19.11

Overview

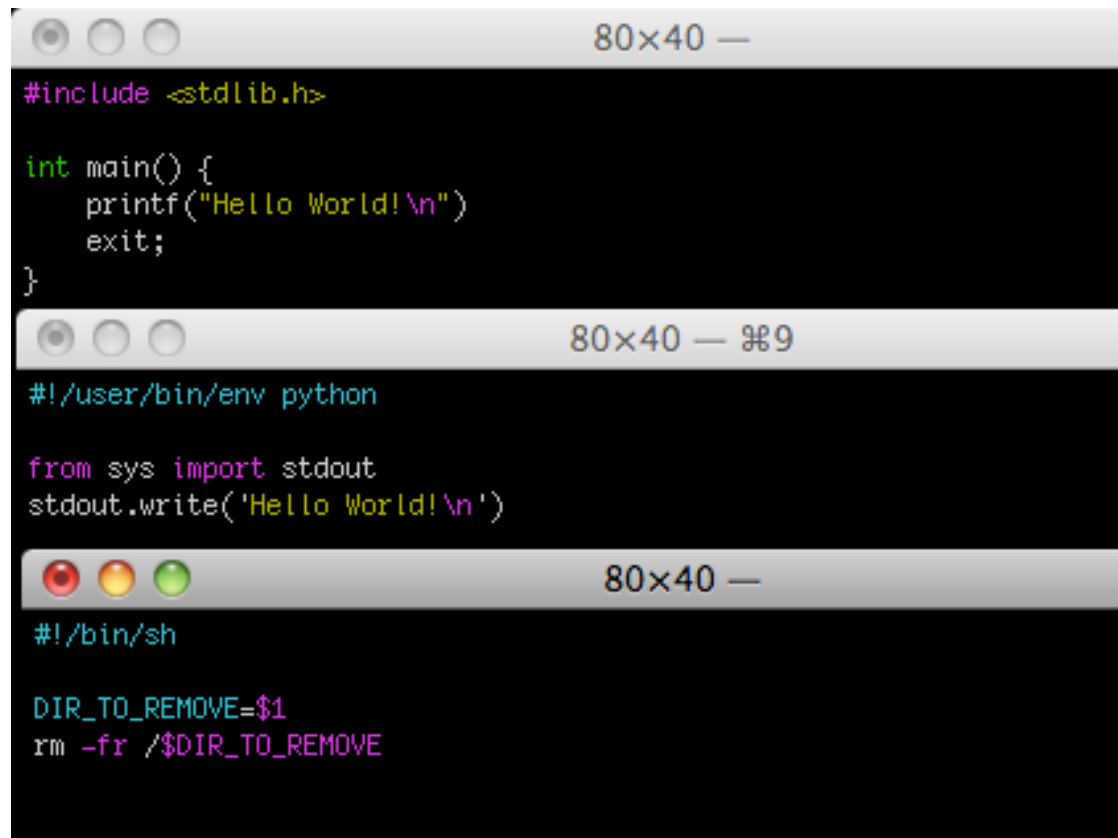
- Discuss testing and why testing is important in software development
- Talk about the tax2tree project in collaboration with Greengenes
- Develop a command line script

Testing

- Software is complex and bugs can be sneaky
- Humans are error-prone, computers are stupid
- There are many types of testing
 - White box
 - Grey box
 - Black box
- There are many types of bugs
 - Arithmetic
 - Logic
 - Performance
- Publications based on untested code is dangerous



Can you spot the bugs?



The image shows three overlapping terminal windows, each with a title bar and a dark background. The top window has a title bar with three buttons and the text '80x40 —'. It contains C code with a bug: `#include <stdlib.h>` (the angle brackets are missing), `int main() {`, `printf("Hello World!\n")`, `exit;` (missing semicolon), and `}`. The middle window has a title bar with three buttons and the text '80x40 — №9'. It contains a Python script with a bug: `#!/user/bin/env python` (the path is incorrect), `from sys import stdout`, and `stdout.write('Hello World!\n')`. The bottom window has a title bar with three colored buttons (red, yellow, green) and the text '80x40 —'. It contains a shell script with a bug: `#!/bin/sh`, `DIR_TO_REMOVE=$1`, and `rm -fr /$DIR_TO_REMOVE` (the path is incorrect).

```
#include <stdlib.h>

int main() {
    printf("Hello World!\n")
    exit;
}
```

```
#!/user/bin/env python

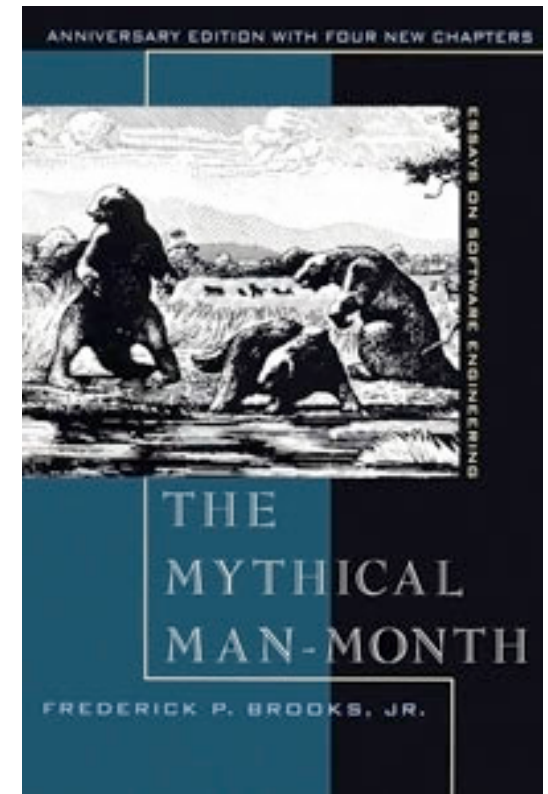
from sys import stdout
stdout.write('Hello World!\n')
```

```
#!/bin/sh

DIR_TO_REMOVE=$1
rm -fr /$DIR_TO_REMOVE
```

Testing

- Software is complex and bugs can be sneaky
- Humans are error-prone, computers are obedient and stupid
- There are many types of testing
 - White box
 - Grey box
 - Black box
- There are many types of bugs
 - Arithmetic
 - Logic
 - Performance
- Publications based on untested code is dangerous



Testing Cont'd

- Unit-testing
 - A unit is a small well defined block of code
 - Tests verify the correctness of these blocks
 - Mocking is encouraged
- Integration testing
 - The whole is greater than the sum of the parts
 - Verify whole workflows
- Regression testing
 - Verify components as software evolves
 - Automated testing

- “Imperfect tests, run frequently, are much better than perfect tests that are never written at all.”
 - Martin Fowler (<http://www.martinfowler.com/articles/continuousIntegration.html#N100F5>)

PyCogent Scripting

- Use the PyCogent template script
- Add support for querying NCBI
- Write the sequences to a file
- Align the sequences and write to a file
- Reverse complement the alignment and write to a file
- Produce a consensus sequence for the alignment and print it to the screen
- Build a tree from the reverse complemented alignment and write it to a file