Computational Structures in Data Science

Lecture 2:
Abstraction and Functions

Berkeley
UNIVERSITY OF CALIFORNIA

Michael Ball | UC Berkeley | https://c88c.org |

Announcements

- Join the EECS 101 and DATA 001 Ed Discussions!
  - https://eecs.link/join-ed
  - https://eecs.link/data-ed
- Hopefully not needed! *Please,* report any concerns about class / campus climate to the department. *You* are welcome here!
- https://eecs.link/climate

Announcements – Waitlist and Exams

- We are working to expand the course.
  - Usually 10-15% people get off the waitlist.
  - This year it keeps growing. ☹
  - **Keep up with the class!**
- Section Signups Released Yesterday
  - Please sign up and attend a regular section
  - Megasection: Useful if you want a little less interactivity.
  - We will track attendance, but not for a grade!
- Exams (reminder):
  - Midterm: Tue October 10
  - Final: Thu Dec 14

Links

- Q&A Thread: https://go.c88c.org/qa2
- Self-Check: https://go.c88c.org/2
- Website Google Calendar: https://c88c.org/fa23/weekly-schedule.html

Computational Structures in Data Science

Abstraction

Berkeley
UNIVERSITY OF CALIFORNIA

Michael Ball | UC Berkeley | https://c88c.org |

Abstraction

- Detail removal

  "The act of leaving out of consideration one or more properties of a complex object so as to attend to others."

- Generalization

  "The process of formulating general concepts by abstracting common properties of instances"

- Technical terms: Compression, Quantization, Clustering, Unsupervized Learning

Henri Matisse *"Naked Blue IV"*

Experiment


Standard Time Zones of the World
WHERE ARE YOU FROM?
CANADA
U.S.
UNITED STATES
MEXICO
BRAZIL
ARGENTINA
CHILE
PERU
BOLIVIA
PARAGUAY
URUGUAY
COLOMBIA
VENEZUELA
ECUADOR
RUSSIA
CHINA
MONGOLIA
KAZAKHSTAN
INDIA
JAPAN
INDONESIA
AUSTRALIA
NEW ZEALAND
PHILIPPINES
ALGERIA
LIBYA
EGYPT
SUDAN
ETHIOPIA
NIGERIA
ZAIRE
SOUTH AFRICA
MADAGASCAR
SAUDI ARABIA
IRAN
IRAQ
TURKEY
FRANCE
GERMANY
UNITED KINGDOM
SPAIN
POLAND
UKRAINE
SWEDEN
NORWAY
FINLAND
North Atlantic Ocean
South Atlantic Ocean
North Pacific Ocean
South Pacific Ocean
Indian Ocean
Coordinated Universal Time (U.T.C.) formerly Greenwich Mean Time
International Date Line
C.A.R. -Central African Republic
F.Y.R.O.M. -The Former Yugoslav Republic of Macedonia

Where are you from?

Possible Answers:

- Planet Earth
- Europe
- California
- The Bay Area
- San Mateo
- 1947 Center Street, Berkeley, CA
- 37.8693° N, 122.2696° W

All correct but different levels of abstraction!

Detail Removal (in Data Science)

- You'll want to look at only the interesting data, leave out the details, zoom in/out…
- Abstraction is the idea that you focus on the essence, the cleanest way to map the messy real world to one you can build
- Experts are often brought in to know what to remove and what to keep!

The London Underground 1928 Map & the 1933 map by Harry Beck.

The Power of Abstraction, Everywhere!

- Examples:
  - Math Functions (e.g., sin x)
  - Hiring contractors
  - Application Programming Interfaces (APIs)
  - Technology (e.g., cars)

- Amazing things are built when these layer
  - And the abstraction layers are getting deeper by the day!

We only need to worry about the interface, or specification, or contract NOT how (or by whom) it's built

Above the abstraction line

Abstraction Barrier (Interface)
(the interface, or specification, or contract)

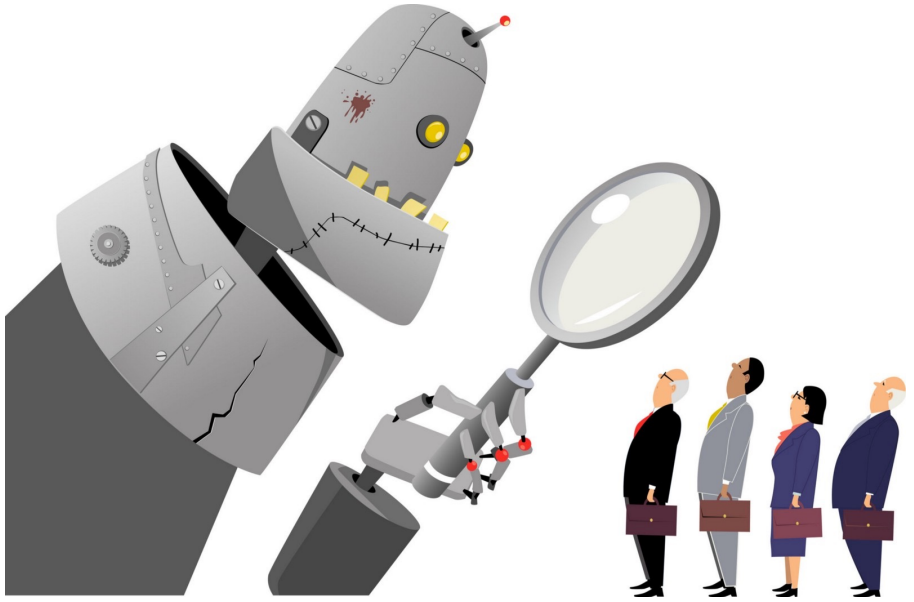
Below the abstraction line

This is where / how / when / by whom it is actually built, which is done according to the interface, specification, or contract.

Abstraction: Pitfalls

- Abstraction is not universal without loss of information (mathematically provable). This means, in the end, the complexity can only be “moved around”

- Abstraction makes us forget how things actually work and can therefore hide bias. Example: AI and hiring decisions.



- Abstractions can formalize a design or pattern. When something doesn't follow that pattern–perhaps a new use

Data or Code? Abstraction→ Take CS61C

Human-readable code (programming language)

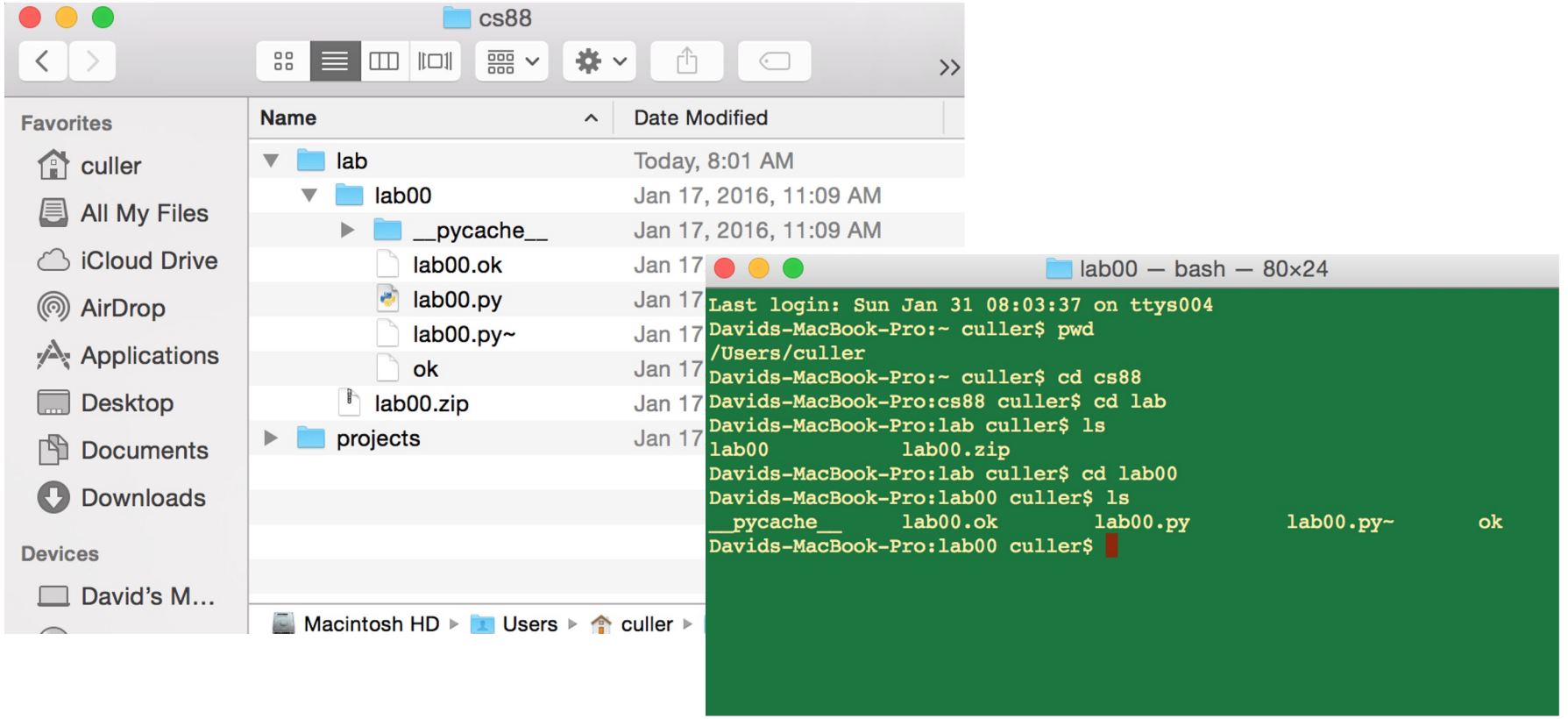
```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodename()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print ' %s [label="%s' % (nodename, label),
    if isinstance(ast[1], str):
        if ast[1].strip():
            print '= %s"];' % ast[1]
        else:
            print '"]'
    else:
        print '"];'
        children = []
        for n, child in enumerate(ast[1:]):
            children.append(dotwrite(child))
        print ' %s -> {' % nodename,
        for name in children:
            print '%s' % name,
```

Machine-executable instructions (byte code)

Compiler or Interpreter

Here: Python

Computers Are Built On Abstractions



- Big Idea: Layers of Abstraction
  - The *GUI* look and feel is built out of files, directories, system code, etc.

Review:

- Abstraction:
  - Detail Removal or Generalizations
- Code:
  - Is an abstraction!

Computer Science is the study (and building) of abstractions

Computational Structures in Data Science

Python: Simple Statements

Berkeley
UNIVERSITY OF CALIFORNIA

Michael Ball | UC Berkeley | https://c88c.org

Learning Objectives

- Evaluate Python Expressions
- Call Functions in Python
- Assign data to Variables

Let's talk Python

- **Expression** `3.1 * 2.6`
- ***Call* expression** `max(0, x)`
- Variables `my_name`
- Assignment Statement `my_name = <expression>`
- **Define Statement:** `def function_name(<arguments>):`
- Control Statements: `if …`
  `for …`
  `while …`

- Comments `# Text after the # is ignored.`

Boolean Expressions

- **Booleans** are Yes/No values.
  - In Python: `True` and `False`
- `>, <, ==, !=, >=, <=, and, or`
  - Note the the *"double equals"*
- These expressions all return only True or False.
- `3 < 5 # returns True`
  - You can write `3 < 5 == True` – but this is redundant.
- We'll keep practicing over time

Python Statements and Expressions

- A *statement* is any particular piece of code
- In an *expression* we care about the return value

```
print('Welcome to C88C!')
course = 'C88C'
print('Welcome to ' + course + '!')
8 * 11
8 + 80

max(88, 61)
len('Berkeley')
```


Live Coding Demo

- Open Terminal on the Mac
- Type `python3`
  - We are now in the "interpreter" and can type code.
- Python runs each line of code as we type it.
  - After each line, we see a result. This happens *only* in the interpreter.
- It's a very useful calculator.
- We can also run files!
- `python3 -i 02-Functions.py`
  - `-i :` This means open the interpreter after running the file. It's optional
- `python3 ok …`
  - This runs the file "ok" which is included with each lab / homework.

Computational Structures in Data Science

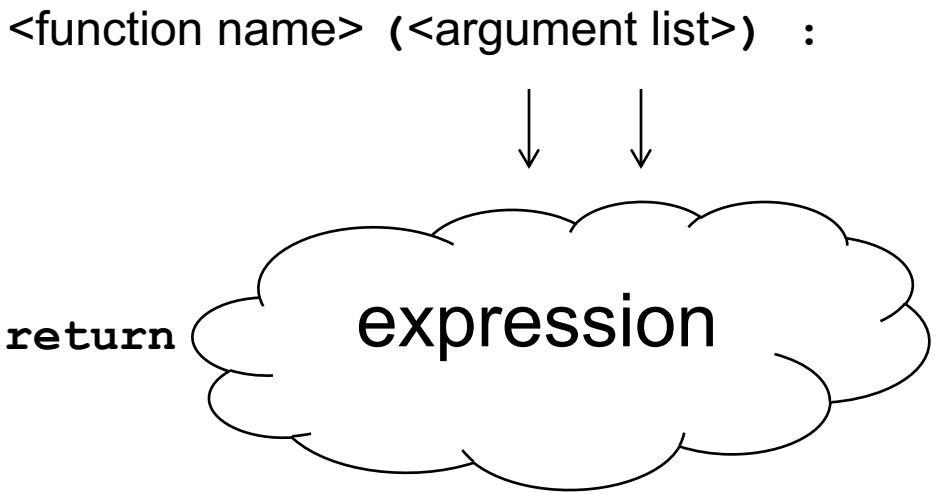
Python: Function Definitions

Berkeley
UNIVERSITY OF CALIFORNIA

Learning Objectives

- Create your own functions.
- Use if and else to control the flow of code.

Defining Functions



- Abstracts an expression or set of statements to apply to lots of instances of the problem
- A function should *do one thing well*

Functions in Python

- We "define" them with `def`
- We typically `name_them_using_underscores` ("Snake case")
- The first line ends in a `:`
- The body is indented by 4 spaces
- Arguments (parameters) create 'names' that exist only in our function
- Most functions will return a value, but some do not.

```
def greet(name):
    print("Hello, " + name)
```

Functions: Example

Let's write a simple function which returns 8 more than the number.

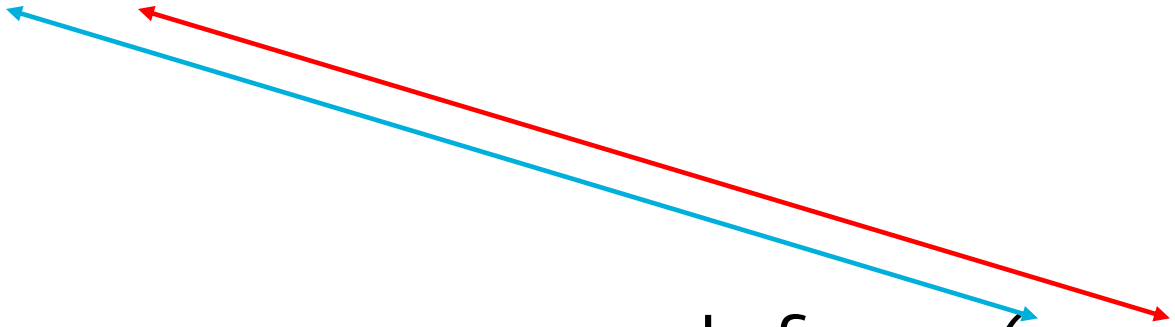
We will *call* this function by writing add_8(80).

Inside, the name num will become the value 80.

```
def add_8(num):
    """add 8 to the input num
    >>> add_8(80)
    88
    """
    return 8 + num
```

Functions: Example

```
>>> y  = 5
>>> x = 3
>>> z = max(3, 5) * 10
>>> z
50
```



```
def max(x, y):
    if x > y:
        return x
    else:
        return y
```

How to Write a Good Function

- Give a descriptive name
  - Function names should be lowercase. If necessary, separate words by underscores to improve readability. Names are extremely suggestive!
- Chose meaningful parameter names
  - Again, names are extremely suggestive.

- Write the docstring to explain *what* it does
  - What does the function return? What are corner cases for parameters? Python Style Guide "PEP 8"

- Write *doctest* to show what it should do
  - Before you write the implementation.

Live Coding Demo

Computational Structures in Data Science

Functions and Environments

Berkeley
UNIVERSITY OF CALIFORNIA

Michael Ball | UC Berkeley | https://c88c.org

Python Tutor

```
def max(x, y):
    return x if x > y else y

x = 3
y = 4 + max(17, x + 6) * 0.1
z = x / y
```

Computational Structures in Data Science

Iteration With While Loops

Berkeley
UNIVERSITY OF CALIFORNIA

Michael Ball | UC Berkeley | https://c88c.org

Learning Objectives

- Write functions that call functions
- Learn How to use while loops.

`while` Statement – Iteration Control

- Repeat a block of statements until a predicate expression is satisfied

```
<initialization statements>
while <predicate expression>:
    <body statements>

<rest of the program>
```