The Art of Data Structures

Python Review

Alan Beadle
CSC 162: The Art of Data Structures

Acknowledgment:
Slides and course materials originally prepared by Richard E Sarkis for the previous offering of this course

UNIVERSITY of ROCHESTER

Agenda

- Review the Python programming language

Review of Basic Python

Review of Basic Python

- This course uses Python exclusively
- If you are unfamiliar with Python, you need to be confident in learning the language ASAP, or else this course will be challenging

Review of Basic Python

- The Interpreter (IDLE, Jupyter or CLI)
- Python code can be run many different ways:
  - Python using the IDLE GUI application
  - Through a terminal (Mac/Linux) or Command Prompt (Windows)
  - Using web-based tools like Jupyter (IPython)
  - Embedded in other applications (like Blender)

Review of Basic Python

- These slides are not meant to be a comprehensive review
- They are a guide for the instructor to demonstrate the topics in class
- Refer to Reading Room on the class website
- You may want to look at some CSC 161 materials: http://www.pas.rochester.edu/~rsarkis/csc161/lectures.html

Data Types review

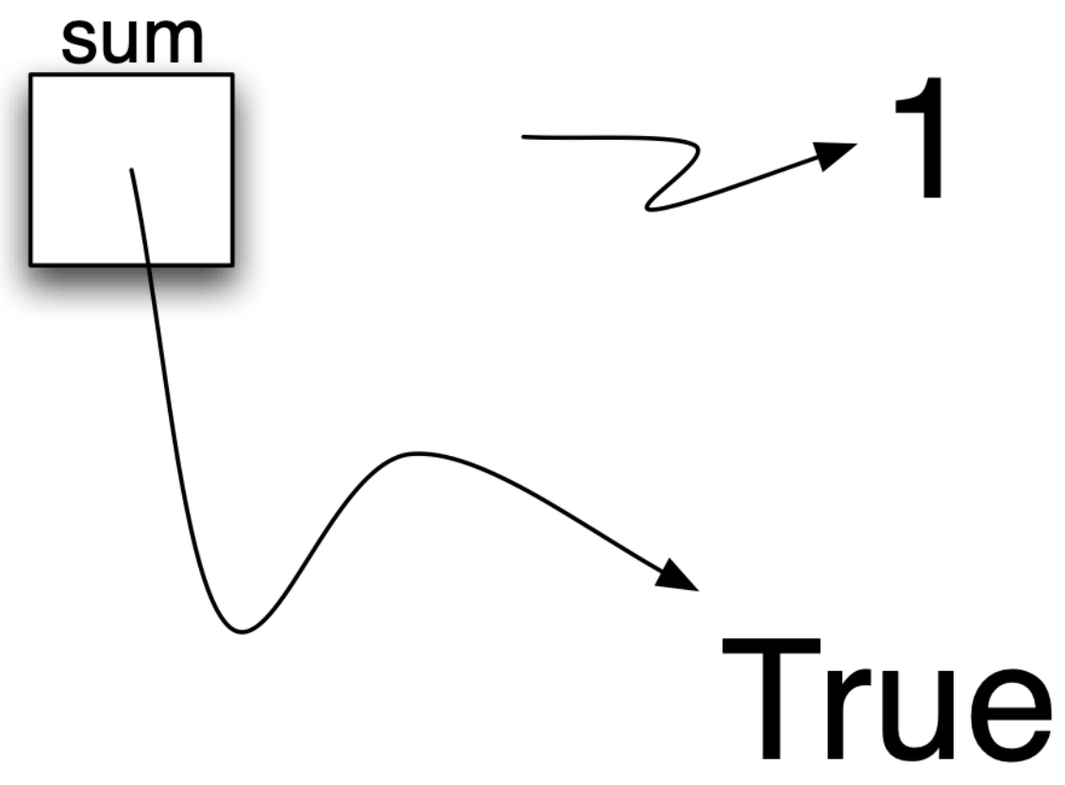
(in notebook)

Variables

Variables

- Variables aren't "buckets" for holding objects
- Rather, they are names pointing to objects

Variables



- A variable (name) can point to any object type
- Changing its value doesn't change the original value

Variables

- We can see this firsthand by checking the id of a variable, modifying the variable, and seeing that the id changes

Control Flow and Exception Handling (in notebook)

Functions

Square Root (Newton's Method)

```
def squareroot(n):
    root = n/2
    for k in range(20):
        root = (1.0/2)*(root + (n /  root))
    return root
```

Classes

PlayingDie Class

```
from random import randrange

class PlayingDie:
    def __init__(self, value=1):
        self.sides = 6
        self.value = value

    def roll(self):
        self.value = randrange(1,7)

    def __repr__(self):
        return f"PlayingDie({self.value})"
```

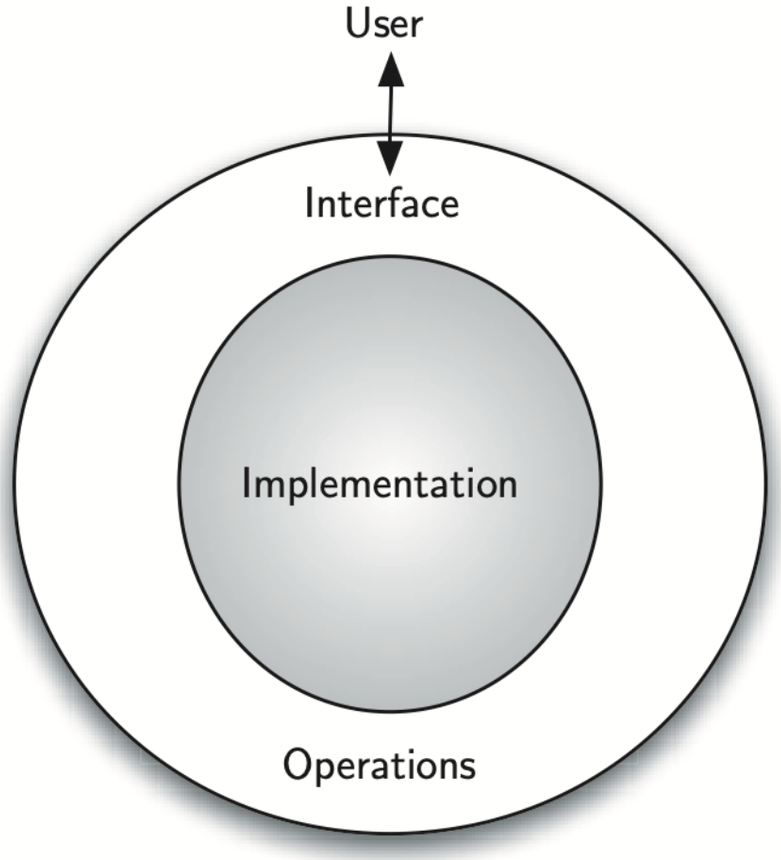
Abstractions

Why Data Structures and Algorithms?

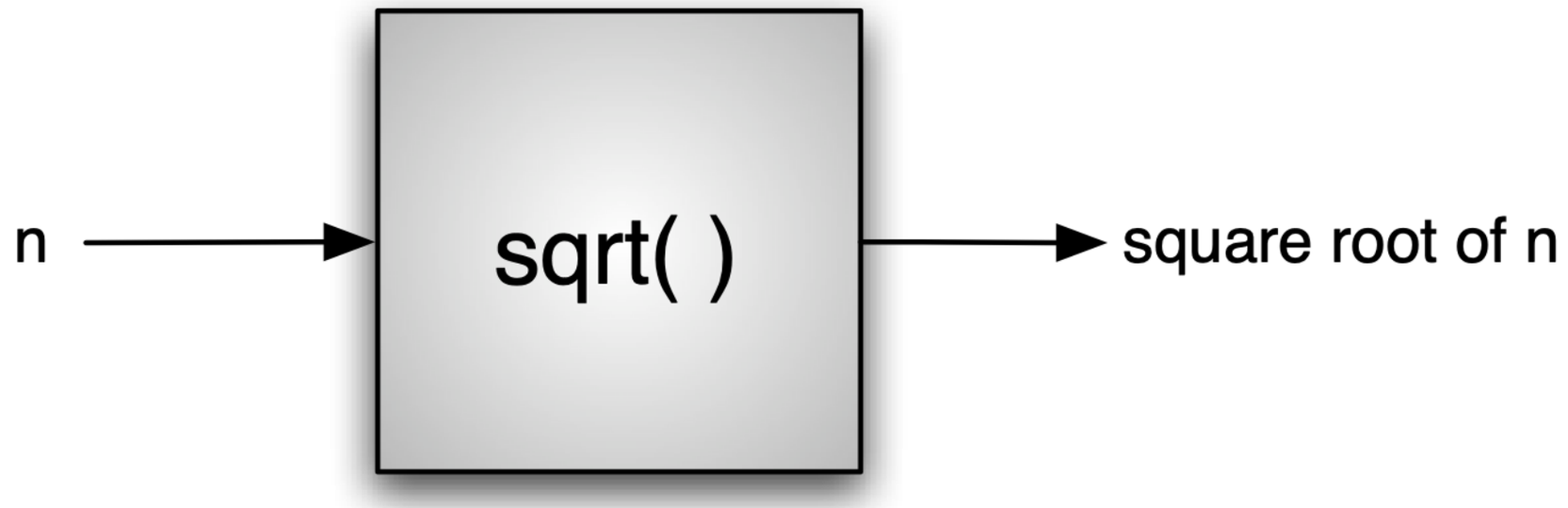
- Abstract data types has similarity to procedural abstraction
- A logical description of how we view the data
- A set of operations allowed
- No care about implementation underneath

Why Data Structures and Algorithms?

- Encapsulation provides information hiding
- Hides the details of the implementation



Procedural Abstraction

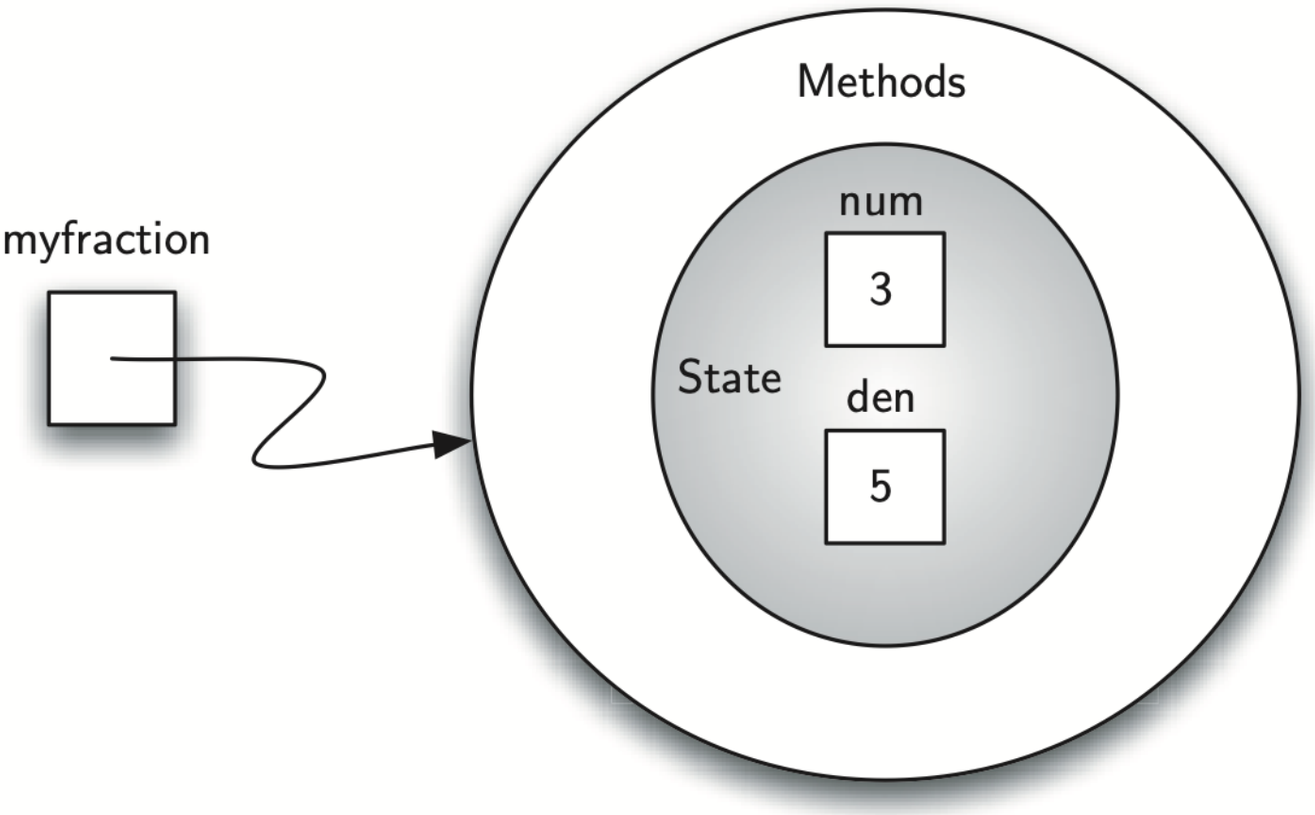


`Fraction` Class

`Fraction` Class (with Initializer)

```
class Fraction:
    def __init__(self, top, bottom):
        self.num = top
        self.den = bottom
```

An Instance of the `Fraction` Class



`Fraction` Class: `show` method

```
def show(self):
    print(self.num,"/",self.den)
```

`Fraction` Class: `__str__` method

```
def __str__(self):
    return str(self.num)+"/"+str(self.den)
```

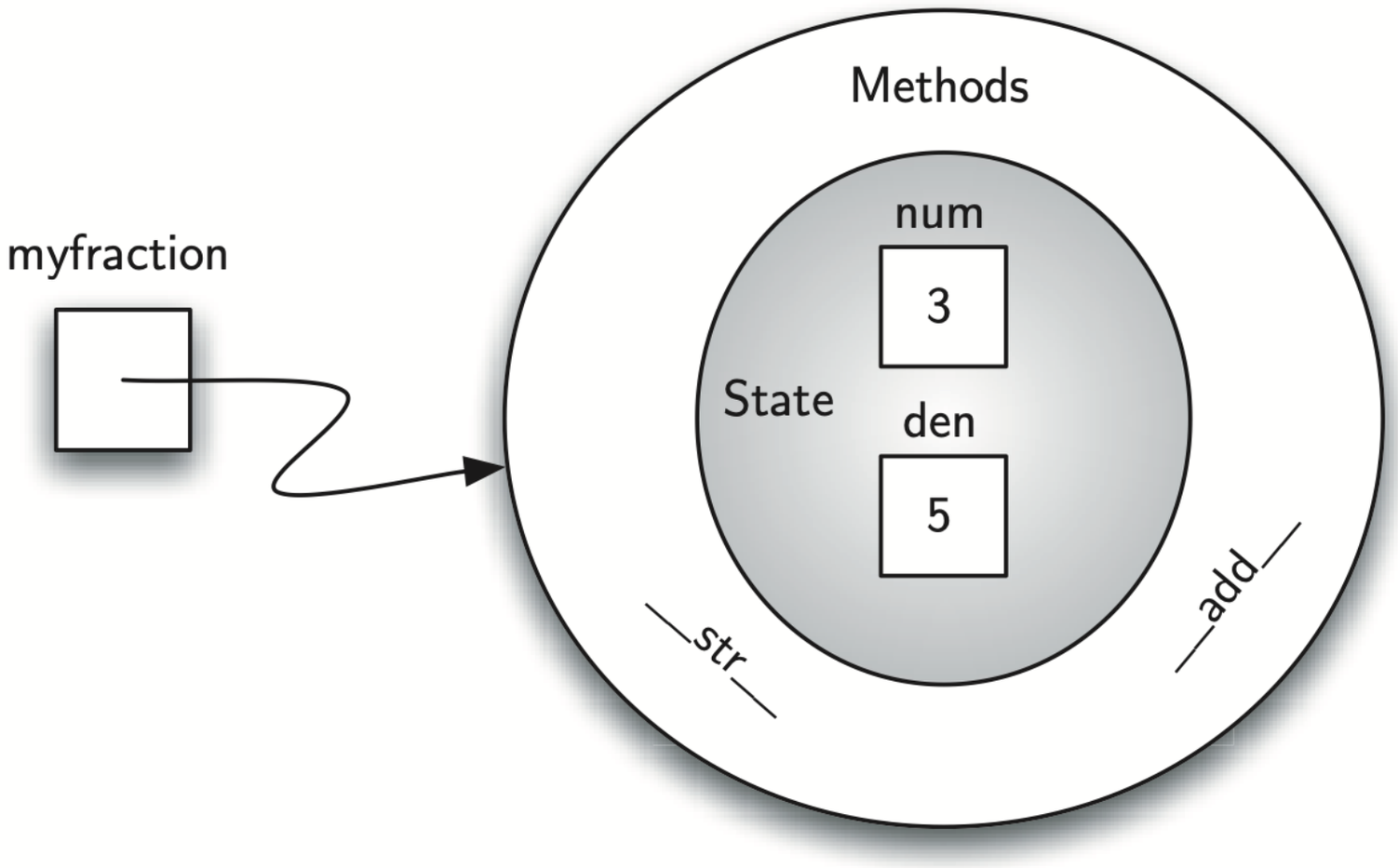
`Fraction` Class: `__add__` method

```
def __add__(self, otherfraction):
    newnum = self.num * otherfraction.den + \
             self.den * otherfraction.num
    newden = self.den * otherfraction.den
    return Fraction(newnum, newden)
```

`Fraction` Class: `gcd` helper function

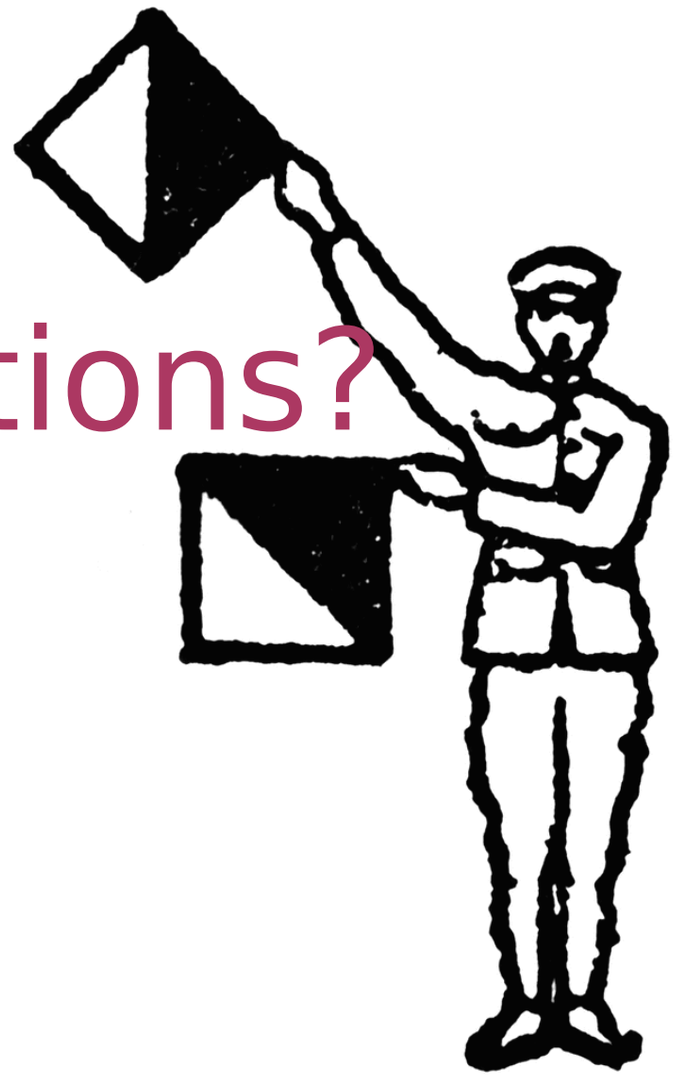
```
# Assume that m and n are
# greater than zero
def gcd(m, n):
    while m%n != 0:
        oldm = m
        oldn = n
        m = oldn
        n = oldm%oldn
    return n
```

`Fraction` Class with two methods



```
class Fraction:
    def __init__(self, top, bottom):
        self.num = top
        self.den = bottom

    def __str__(self):
        return str(self.num)+"/"+str(self.den)

    def show(self):
        print(self.num,"/",self.den)

    def __add__(self, otherfraction):
        newnum = self.num*otherfraction.den + \
                 self.den*otherfraction.num
        newden = self.den * otherfraction.den
        common = gcd(newnum,newden)
        return Fraction(newnum//common,newden//common)

    def __eq__(self, other):
        firstnum = self.num * other.den
        secondnum = other.num * self.den

        return firstnum == secondnum
```

Questions?



- Remember, first assignment appears in blackboard at noon, due in 1 week

- Assignment involves making some changes to the textbook Fraction class.