Presentation 20

Operators and Abstraction

Announcements for Today

Assignments

- A4 graded by Tomorrow
  - Will show survey Thursday
- A5 is due **Today**
  - Graded by **Saturday**
  - Posted in time for exam
- Need to be working on A6
  - Follow **micro-deadlines**
  - Should be on `Filter` now!
  - It is due on **Sunday**

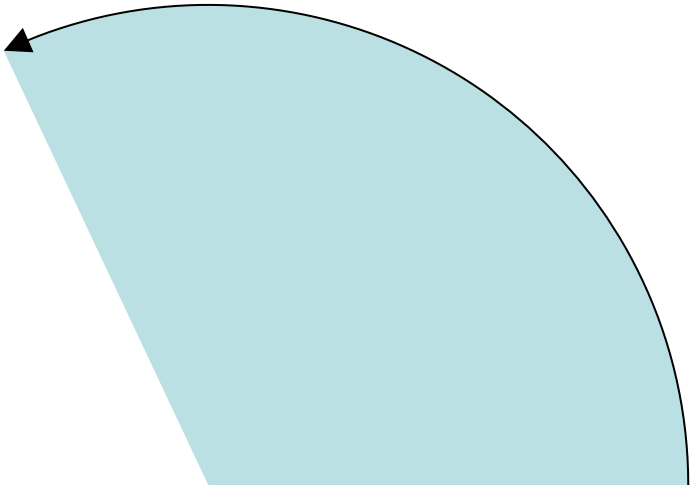
Lessons/Prelim

- **Videos 23.1-23.7** for today
- **Lessons 24, 25** for next time
- **Prelim 2 is Nov 19th**
  - This time at 9:30 am
- **Material up to Thurs**
  - Study guide is now posted
  - Recursion + Loops + Classes
- **Submit conflicts today!**

Case Study: Angles

Degrees: 127.67

Radians: 2.228261856



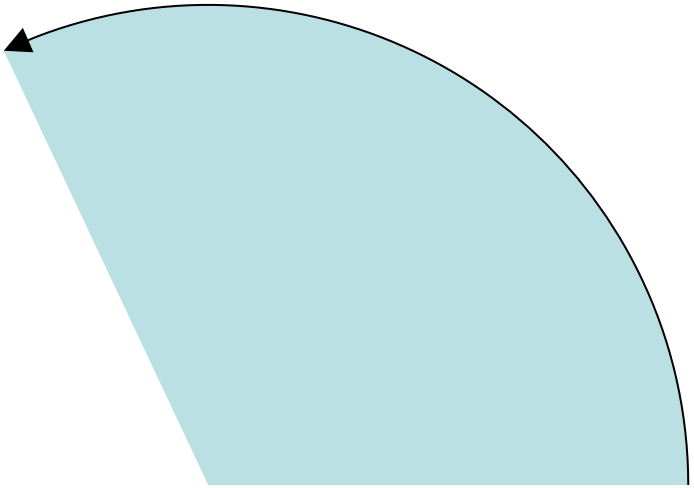
There are many ways to measure

Case Study: Angles

Degrees: 127.67

Radians: 2.228261856

DMS: 127° 40' 12"



There are many ways to measure

The Class Specification

```
class Angle(object):
    """A class representing an angle in DMS format

    The class does not allow a finer grained measurement than seconds
    (e.g. microseconds). All of the values must be integral."""
    # Attribute _degrees: The angle in degrees
    # Invariant: _degrees is (any) int
    # Attribute _minutes: Part of single degree
    # Invariant: _minutes is an int 0..59
    # Attribute _seconds: Part of single minute
    # Invariant: _seconds is an int 0..59
```

Complete the Initializer

```
class Angle(object):
    ...
    def __init__(self, ???):
        """Initializes a new angle

        Parameter d: The number of degrees
        Precondition: d is an int

        Parameter m: The number of minutes (OPTIONAL; default 0)
        Precondition: m is an int 0..59

        Parameter s: The number of seconds (OPTIONAL; default 0)
        Precondition: s is an int 0..59"""
        pass
```

Complete the Initializer

```
class Angle(object):
    ...
    def __init__(self, ???):
        """Initializes a new angle

        Parameter d: The number of deg
        Precondition: d is an int

        Parameter m: The number of mi
        Precondition: m is an int 0..59

        Parameter s: The number of sec
        Precondition: s is an int 0..59"""
        pass
```

What are params?

A: (d,m,s)

B: (self,d,m,s)

C: (self,d,m,s=0)

D: (self,d,m=0,s=0)

E: Unsure

Complete the Initializer

```
class Angle(object):
    ...
    def __init__(self, ???):
        """Initializes a new angle

        Parameter d: The number of deg
        Precondition: d is an int

        Parameter m: The number of mi
        Precondition: m is an int 0..59

        Parameter s: The number of sec
        Precondition: s is an int 0..59"""
        pass
```

What are params?

A: (d,m,s)

B: (self,d,m,s)

C: (self,d,m,s=0)

D: (self,d,m=0,s=0)

E: Unsure

String Representation

```
class Angle(object):
    ...
    def __str__(self):
        """Returns an informal string representation of this angle

        The string has the form d° m' s"   """
        pass

    def __repr__(self):
        """Returns a formal string representation of this angle

        The string has the form Angle[d° m' s"]     """
        pass
```

String Representation

```
class Angle(object):
    ...
    def __str__(self):
        """Returns an informal string representation of this angle

        The string has the form d° m' s"   """
        pass

    def __repr__(self):
        """Returns a f                    ion of this angle

        The string has the                Angle[d° m' s"]     """
        return str(self.__class__)+'['+str(self)+']'
```

Usually the easier one of the two

String Representation

```
class Angle(object):
    ...
    # def __str__(self):
    #     """Returns an informal strin
    #
    #     The string has the form d° m
    #     pass

    def __repr__(self):
        """Returns a formal string rep
        The string has the form Angle
        return str(self.__class__)+'['+
```

Comment out str

```
>>> a = Angle(5,4,3)
>>> repr(a)
```

What is result?

A: 'Angle[5° 4' 3"]'

B: 'Angle[None]'

C: **Crash** (__str__ not found)

D: **Crash** (RecursionError)

E: Unsure

String Representation

```
class Angle(object):
    ...
    # def __str__(self):
    #     """Returns an informal strin
    #
    #     The string has the form d° m
    #     pass

    def __repr__(self):
        """Returns a formal string rep
        The string has the form Angle
        return str(self._class)+'['+str(
```

Comment out str

```
>>> a = Angle(5,4,3)
>>> repr(a)
```

What is result?

A: 'Angle[5° 4' 3"]'

B: 'Angle[None]'

C: **Crash** (__str__ not found)

D: **Crash** (RecursionError)

E: Unsure

Basic Arithmetic

```
class Angle(object):
    ...
    def __add__(self,value):
        """Returns an angle that the sum of this angle and value

        Parameter value: The angle to add
        Precondition: value is an Angle"""
        pass

    def __sub__(self,value):
        """Returns an angle that the difference of this angle and value"""
        ...
```

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that is the multiple of this angle and value

        Parameter value: The value to multiply by
        Precondition: value is ?????"""
        pass
```

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that
        Parameter value: The val
        Precondition: value is ??
        pass
```

What is precondition?

A: value an Angle

B: value an int

C: value a number

D: value a float

E: Unsure

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that
        Parameter value: The val
        Precondition: value is an
        pass
```

What is precondition?

A: value an Angle

B: value an int

C: value a number

D: value a float

E: Unsure

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that is

        Parameter value: The value
        Precondition: value is an in
        pass
```

```
>>> a = Angle(5,29,33)
>>> a*2
```

What is result?

A: 10° 29' 33"

B: 10° 58' 66"

C: 10° 59' 6"

D: **ERROR**

E: Unsure

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that is

        Parameter value: The value
        Precondition: value is an in
        pass
```

```
>>> a = Angle(5,29,33)
>>> a*2
```

What is result?

A: 10° 29' 33"

B: 10° 58' 66"

C: 10° 59' 6"

D: **ERROR**

E: Unsure

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that is

        Parameter value: The value
        Precondition: value is an in
        pass
```

```
>>> a = Angle(5,29,33)
>>> 2*a
```

What is result?

A: 10° 29' 33"

B: 10° 58' 66"

C: 10° 59' 6"

D: **ERROR**

E: Unsure

A Little More Interesting

```
class Angle(object):
    ...
    def __mul__(self,value):
        """Returns an angle that is

        Parameter value: The value
        Precondition: value is an in
        pass
```

```
>>> a = Angle(5,29,33)
>>> 2*a
```

What is result?

A: 10° 29' 33"

B: 10° 58' 66"

C: 10° 59' 6"

D: **ERROR**

E: Unsure

And Now it Gets Harder

```
class Angle(object):
    ...
    def __???__(self,value):
        """Returns an angle that

        Parameter value: The val
        Precondition: value is an
        pass
```

What is operator?

A: __truediv__

B: __floordiv__

C: __divmod__

D: There isn't one

E: Unsure

And Now it Gets Harder

```
class Angle(object):
    ...
    def __???__(self,value):
        """Returns an angle that
        Parameter value: The val
        Precondition: value is an
        pass
```

What is operator?

A: __truediv__

B: __floordiv__

C: __divmod__

D: There isn't one

E: Unsure

To Make Division Easier

```
class Angle(object):
    ...
    @classmethod
    def forDegrees(cls,d):
        """Returns the angle for the given degrees as a decimal

        Because seconds must be an int, there is no guarantee that we can make
        an angle equal to d. The angle will be the floor,  which is the largest
        angle less than or equal to d

        Parameter d: The decimal degrees
        Precondition: d is a float"""
        pass
```

And One Last Thing

```
class Angle(object):
    ...
    def __eq__(self,value):
        """Returns true of value is an Angle equals to this one

        Parameter value: The object to compare
        Precondition: ????"""
        pass
```

And One Last Thing

```
class Angle(object):
    ...
    def __eq__(self,value):
        """Returns true of va

        Parameter value: Th
        Precondition: ????"""
        pass
```

What is precondition?

A: value an Angle

B: value an int

C: value a number

D: value can be **anything**

E: Unsure

And One Last Thing

```
class Angle(object):
    ...
    def __eq__(self,value):
        """Returns true of va

        Parameter value: Th
        Precondition: ????"""
        pass
```

What is precondition?

A: value an Angle

B: value an int

C: value a number

D: value can be **anything**

E: Unsure

Questions?