Objectives

- Designing our own classes
  - Representing attributes/data
  - What functionality to provide
- Using our defined classes

1

Review

- What did yesterday's lab bring together?
  - What were some things you practiced?
- If I gave you a file of all the names from the US Census in the correct form, how much code would you need to change to process/graph the most common names?
- How long did it take the computer to write the outputs of all four files (as compared to printing)?
  - May not be as clear because the printed output is squeezed
- Why classes and objects?
- How do we create new data types?

2

Parts of an Algorithm

- Input, Output
- Primitive operations
  - What data you have, what you can do to the data
- Naming
  - Identify things we're using
- Sequence of operations
- Conditionals
  - Handle special cases
- Repetition/Loops
- Subroutines
  - Call, reuse similar techniques

Where We Are

- With what you now know, 10 weeks in
  - Opens up the possibilities for what you kinds of programs you can write
  - Just about anything computational is possible

Parts of an Algorithm

- Input, Output
- Primitive operations
  - What data you have, what you can do to the data
- Naming
  - Identify things we're using
- Sequence of operations
- Conditionals
  - Handle special cases
- Repetition/Loops
- Subroutines
  - Call, reuse similar techniques

Going beyond the primitive data to making our own structures

Example: Student – for the Registrar

- Data to model for a Student?
- API for a Student?

Student Class

- Each student has these **attributes**:
  - First name
  - Last name
  - Expected graduation
  - Majors
  - Minors

Students all have these attributes, but different values for the attributes

Each student is an **instance of** the `Student` class

- Methods
  - getExpectedGraduationYear()
  - getFirstName()
  - declareMajor(major)

Review: Object-Oriented Programming

- Defining a class
  - Why do we want to define classes/new data types?
  - What is the keyword to create a new class?
  - How do you define a method?
    - What parameter is needed in every method?
      - What does that parameter represent?
  - How do we access instance variables in methods?
- Using a class
  - How do you create a new object of a given class?
    - What method does this call?
  - How do you call a method?
  - What method is called when you print an object?
    - What is that method's signature?

Card Class (Incomplete)

Class Doc String

```
class Card:
    """ A class to represent a standard playing card.
    The ranks are ints: 2-10 for numbered cards, 11=Jack,
    12=Queen, 13=King, 14=Ace.
    The suits are strings: 'clubs', 'spades', 'hearts',
    'diamonds'."""

    def __init__(self, rank, suit):
        """Constructor for class Card takes int rank and
         string suit."""
        self._rank = rank
        self._suit = suit

    def getRank(self):
        "Returns the card's rank."
        return self._rank

    def getSuit(self):
        "Returns the card's suit."
        return self._suit
```

Method Doc String

Methods

Methods are like *functions* defined in a *class*

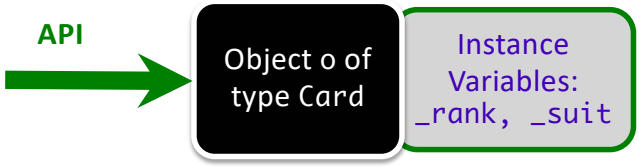
Example: Rummy Value

- Problem: Add a method to the `Card` class called `getRummyValue` that returns the value of the card in the game of Rummy
- Procedure for defining a method (similar to functions)
  - What is the input to the method?
  - What is the output from the method?
  - What is the method signature/header?
  - What does the method do?
- How do we call the method?
- How can we test the method?

Card API

- Based on what we've seen/done so far, what is the Card class's API?
  - (Review: What is an API?)

Card API



- `Card(<rank>, <suit>)` Implementation of methods is hidden
- `getRank()`
- `getSuit()`
- `getCardColor()`
- `getRummyValue()`
- `__str__()` or `str(card)`

Algorithm for Creating Classes

1. Identify need for a class
2. Identify state or attributes of a class/an object in that class→ these are the *instance variables*
3. Write the constructor (`__init__`)
   - Initialize the instance variables
4. Implement the `__str__` method
   - Test the `__str__` method
5. Identify methods the class should provide
   - How will a user call those methods (parameters, return values)?
     - Develop API
   - Implement and test methods one at a time

Using the `Card` class

- Having the `Card` class means that we can represent a `Card` in code

Now that we have the `Card` class, how can we **use** it?

Using the Card class

Now that we have the Card class, how can we **use** it?

- Calculate the score of your hand in Rummy

Left as an exercise to practice

Exam 2 Questions

- Content
  - Everything up through dictionaries
    - (Not creating our own classes)
  - Necessarily cumulative but focus is on second half
- What types of questions are you expecting?

Exam 2

- What operations/methods can you do on strings? Lists? Dictionaries? Files?
- What is the syntax for writing conditions? Conditional statements? while loops?
- Problems:
  - Modify your lab 9 code to find the x^{th} most popular name
  - Modify the first lab 9 problem to take a string and, for each word in the string, maps the first letter to the word (like a children's book)
    - Other modifications: read from a file; make a function that returns the dictionary