Dictionaries

CSSE 120—Rose Hulman Institute of Technology

Data Collections

- Frequently several individual pieces of data are related
- We can collect them together in one object
- Examples:
  - A list or tuple contains an ordered sequence of items
  - A string contains an ordered sequence of characters
  - A custom object. Example from zellegraphics: A Line object contains two endpoints, a color, and the window in which it is drawn
  - A dictionary (defined soon) contains key-value pairs

List - review

- an ordered collection of items
- Usually homogeneous (all items of the same type), but Python does not require this
- Access is **by position** (index) in the list

```
>>> animals = ['dog', 'cat', 'cow']
>>> animals[1]
'cat'
>>> animals[1:3]
['cat', 'cow']
>>> animals[1] = ['pig']
>>> animals
['dog', 'pig', 'cow']
```

More list mutations

- Items can be added, removed, or replaced

```
>>> animals = ['dog', 'cat', 'cow']
>>> animals.append('pig')
>>> animals
['dog', 'cat', 'cow', 'pig']
>>> animals[1:3] = ['cow', 'cat', 'goat']
>>> animals
['dog', 'cow', 'cat', 'goat', 'pig']
>>> animals[1:2] = []
>>> animals
['dog', 'cat', 'goat', 'pig']
```

Dictionary

- A collections object in which each item is a key-value pair
- No two items may have the same key
  - So a dictionary is a function (in the mathematical sense)
- Items are not stored in any particular order
- Typically all keys are same type (not required)
- Keys must be immutable (i.e., number, string, tuple)
- Access to items is by key
  - key's purpose is similar to list's index
  - syntax also similar

Your turn

- Open IDLE and make a quick dict
- Try the following:

```
>>> myDict = {'name':'Dave', 'gpa':3.5}
>>> print myDict
>>> myDict['name']
>>> myDict['gpa']
>>>dir(dict)
```

Dictionary methods

Assume that there is a dictionary named dict1

- dict1.get(k [,d]) → if k is a **key** in the dictionary return the **value** for that key, else return d. d is an optional parameter
- dict1.has_key(k) → True if dict1 has a **key** k, else False
- dict1.items() → list of dict1's (key, value) pairs, as tuples
- dict1.keys() → list of dict1 's **keys**
- dict1.pop(k [,d]) → remove **key** and return **value**
- dict1.values() → list of dict1 's **values**
- Checkout **Session16Dictionaries &** open dictionaryMethods.py

Another dictionary example

```
gradeLowestScore = { } # empty dictionary
gradeLowestScore['A'] = 89.5
gradeLowestScore['B+'] = 84.5
gradeLowestScore['B'] = 79.5
gradeLowestScore['C+'] = 74.5
gradeLowestScore['C'] = 69.5
gradeLowestScore['D+'] = 64.5
gradeLowestScore['D'] = 59.5
gradeLowestScore['F'] = 0.0
```

```
difference = gradeLowestScore['B']-
             gradeLowestScore['C']
```

Q2

dict initialization & operations

```
>>> gradeLowestScore = {'A':89.5, 'B+':84.5, 'B':79.5,
          'C+':74.5, 'C':69.5, 'D+':64.5, 'D': 59.5, 'F': 0.0}
>>> gradeLowestScore['C']
69.5
>>> gradeLowestScore['C'] = 68.0  # new value for key 'C'
>>> gradeLowestScore.keys()
['A', 'C+', 'C', 'B', 'D+', 'F', 'D', 'B+']
>>> gradeLowestScore.values()
[89.5, 74.5, 68.0, 79.5, 64.5, 0.0, 59.5, 84.5]
>>> gradeLowestScore.items()
[('A', 89.5), ('C+', 74.5), ('C', 68.0), ('B', 79.5), ('D+',
64.5), ('F', 0.0), ('D', 59.5), ('B+', 84.5)]
>>> gradeLowestScore.pop('C')  # remove 'C' item
68.0
>>> 'C' in gradeLowestScore
False
>>> 'D' in gradeLowestScore
True
```

dict's *get* method

- What if we try to find the lowest score for an "E" grade?
-
```
>>> gradeLowestScore['E']
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    gradeLowestScore['E']
KeyError: 'E'
```
- The **get** method has a similar purpose, but lets us provide a value to return if the key we search for is not in the dictionary:
-
```
>>> gradeLowestScore.get('E', 'No such key')
'No such key'
```

Two main dictionary uses

- A collection of similar objects
  - Designed for fast lookup by key
- Storing different properties of a single object

Use 1: Collection of similar objects

- Examples:
  - A movie database in which we use the title as the key and look up the director.
  - A phone database in which we use the person's name as the key and look up the phone number
- In-class exercise
  - Create a concordance for a text file.
  - This is just a list of words in the file and the line numbers on which each word occurs

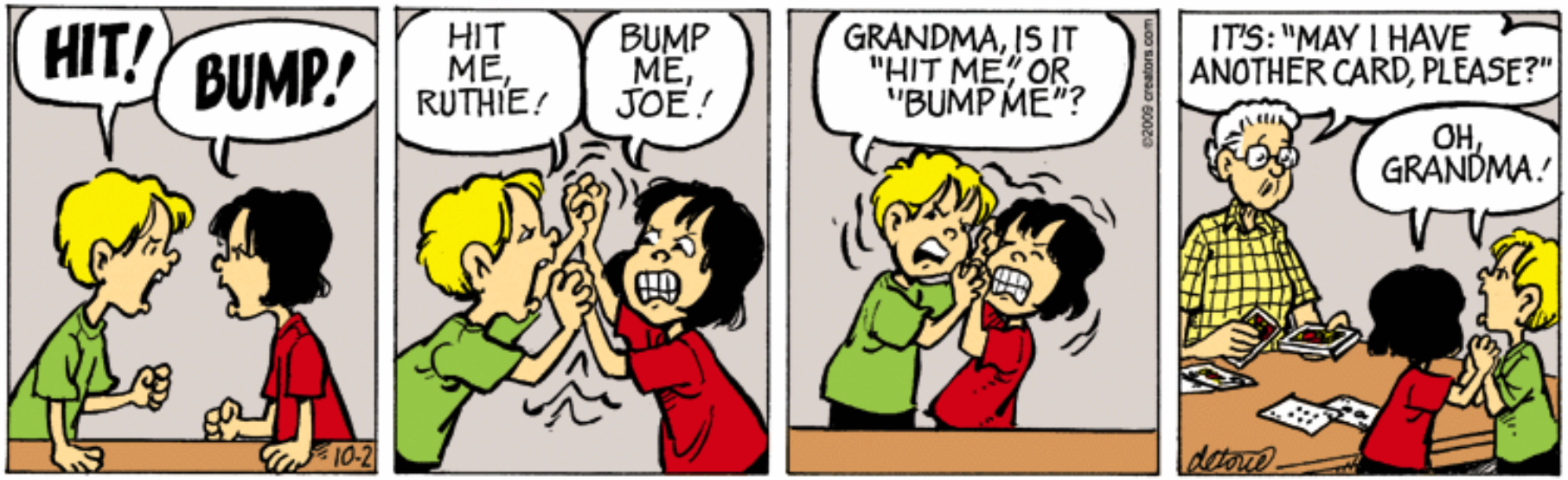
Towards a movie "database"

- Create a dictionary of movie directors:

```
director = {}
director['Star Wars'] = 'George Lucas'
director['The Godfather'] = 'Francis Ford Coppola'
director['American Graffiti'] = 'George Lucas'
director['Princess Bride'] = 'Rob Reiner'

print director['Star Wars'], 'directed', 'Star Wars.'
```

This example is adapted from *Object-Oriented Programming in Python* by Michael Goldwasser and David Letscher


HIT!
BUMP!
10-2
HIT ME, RUTHIE!
BUMP ME, JOE!
GRANDMA, IS IT "HIT ME" OR "BUMP ME"?
©2009 creators.com
IT'S: "MAY I HAVE ANOTHER CARD, PLEASE?"
OH, GRANDMA!
detorie

Use 2: Properties of a single object

- Represent a card (blackjack) as a dictionary
- properties: 'cardName', ' suit', ' value'

```
# A card is represented by a dictionary with keys
# cardName, suit, and value

def makeCard (cardName, suit):
    card = {}
    card['suit'] = suit
    card['cardName'] = cardName
    card['value'] = cardValue(cardName)
    return card
```

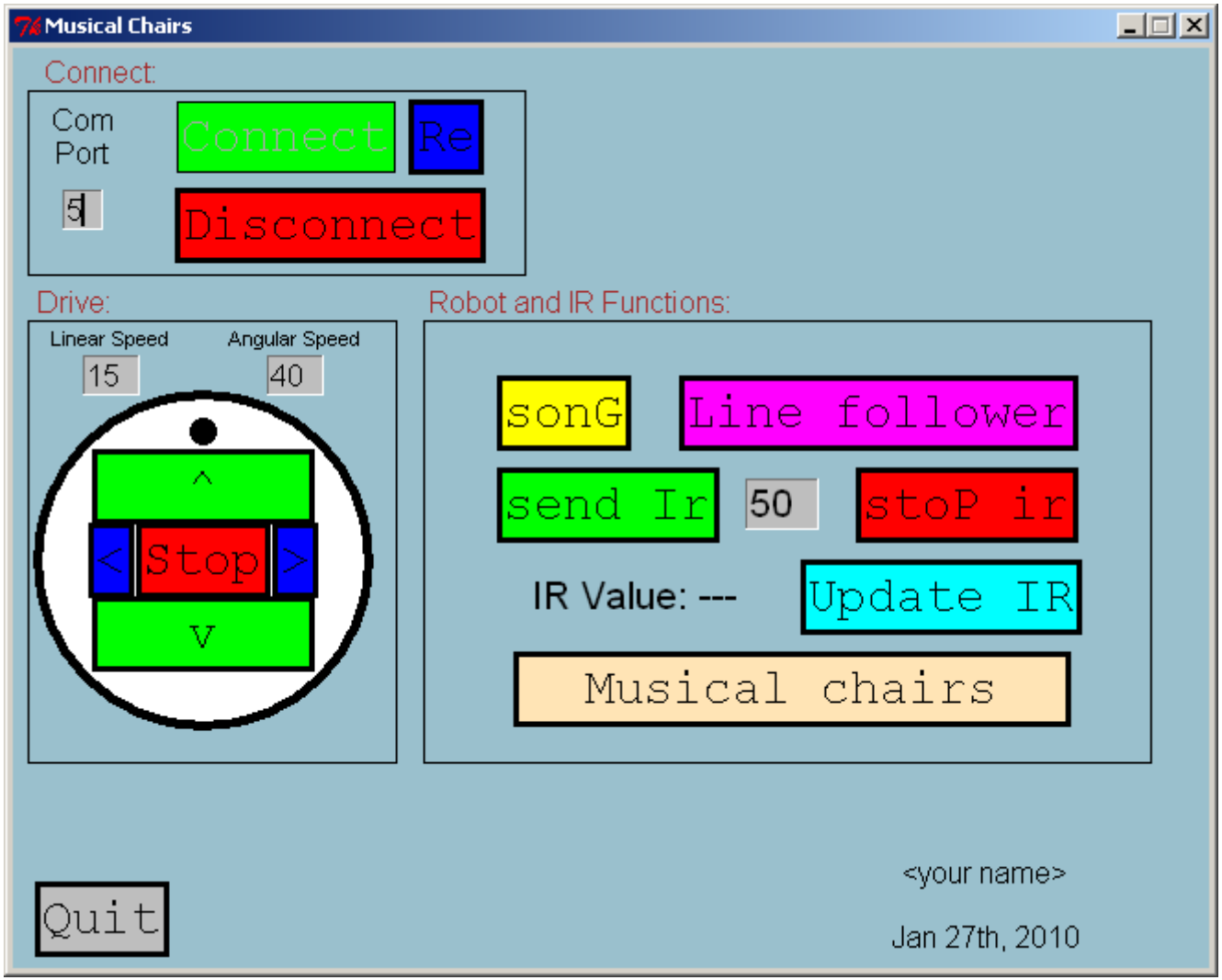
PROJECT KICKOFF

CSSE 120—Rose Hulman Institute of Technology

Project idea

- We will be make a GUI to control the robot
- One of the buttons will be to Play Musical Chairs
  - While receiving the signal 50 to robot go around the line
  - While receiving the IR signal 100 the robot must stop
  - One the robot receives the IR signal 200 the robot should be the first to reach the “chair” the chair is the teacher robot in the middle of the circle
  - There will be a little musical chairs tournament

GUI Layout


Musical Chairs
Connect:
Com Port
5
Connect
Re
Disconnect
Drive:
Linear Speed
15
Angular Speed
40
^
<
Stop
>
v
Robot and IR Functions:
sonG
Line follower
send Ir
50
stoP ir
IR Value: ---
Update IR
Musical chairs
Quit
<your name>
Jan 27th, 2010

Project process

- We will begin the project today
- Due: presentations will be done in Session 20
- Milestones each day along the way
- Today in class: Planning, and deciding what you will have done for each class day

- Exam 2 is Thursday, Jan 28, 7:00 PM

Interaction with your project team

- Brainstorm
  - What to do
  - What not to do

Working with your project team

- Use good practices of **pair-programming,** but with two navigators.
- Have one navigator double as a *facilitator* to make sure the team **stays on task.**
- **Rotate** who drives (types the code)
- Give driving **preference** to those with less programming experience, so they can set the pace
- **Encourage the driver**
- **Make sure** the navigators understand the code added so no one gets lost
- **Work by consensus,** not command: don't "take over" the project and do it solo.

Get going

- Meet your teammates
- Exchange contact info
- Agree on when you will meet next (at least one meeting before the weekend)
- Read the assignment (and follow the links). Ask questions on things you do not understand.
- Draw your ideas of what your screen layout will look like
  - Use a whiteboard if you wish
- Think (and write) about object types (dictionaries) that you will need – what will the keys be?
- Figure out and record your milestones. What will you have done before each class day.
- High-level plans before you begin coding
  - Add your notes on all of this to your project and commit to your team repository

Teams

csse120-201020-team31,wangg,alrisham,seligja

csse120-201020-team32,bristokb,bellbw,casimist

csse120-201020-team33,montgomj,armacoce,pedzindm

csse120-201020-team34,schletal,fennigea,masonkd

csse120-201020-team35,patelms,mcgeevsa,powellcl

csse120-201020-team36,breenjw,verkamma,webbjd

csse120-201020-team37,solorzaa,whitemrj,teemanem

csse120-201020-team38,gibsonjr,goodwiba,kovacidw

Open the SVN Repositories tab, right-click, and choose New → Repository Location...

In the "Add SVN Repository" window, type:

http://svn.cs.rose-hulman.edu/repos/csse120-201020-teamXX

Project Location

- ANGEL → Lesson → Projects → MusicalChairs
- Also linked from Session 16 on the Schedule page, so you do not need to go through ANGEL at all.
- Be sure to read the linked articles and demos.