

## Plan for the Week

- Understand programming in general in Python
  - How do we design programs?
    - What is a program?
  - How do we design functions?
  - How do we design modules?
- Review lists, loops, functions, understand sets
  - Motivate with examples that are connected to the world
  - Sometimes the connection is tenuous
  - What do you do when you have free time?

CompSci 6/101, Spring 2012

7.1

## How to write a program (in Python)

- Understand the big picture of what you're doing
  - For small programs this is likely manageable
  - For big programs, the big picture might be very big
- What are the *use-cases*, different *granularities*
  - How will a user interact with the program or system?
  - Small grain-size to larger-grain size

*User will choose which transform to use*  
*User will be able to specify a filename for saving*
- We will do this informally, often done formally as part of specifying what the program/system does

CompSci 6/101, Spring 2012

7.2

## Getting a program to work

- Write small functions, test them
  - Each function should do one thing
  - Ideally testable independently of other functions/code
- Develop the program incrementally
  - Write small sections, run them, try them, test them
  - Make it run, make it right, (make it fast, make it small)
- Carefully document function dependencies
  - What parameters needed? What value(s) returned?
  - Side effects altering parameters or program state
    - e.g., printing values or saving a file

CompSci 6/101, Spring 2012

7.3

## 31 U.S.C. § 5361–5367

- US Code section 5361 aka UIGEA
  - Unlawful Internet Gambling Enforcement Act
  - Passed in 2006, took effect June 1, 2010
  - What is legal, what is illegal, what are effects?



CompSci 6/101, Spring 2012

7.4

## Statistical Analysis of Poker Hand

- How do we represent cards? Deck? Suit? Rank?

- What is a card?
- What can you do with a card?
- How will we represent a hand?
- Keep things simple: lists help!



7.5

- How do we 'create' a deck

- Number of cards?
- Code for creating cards?
- Loop over suits/ranks
- Debugging assistance!

CompSci 6101, Spring 2012

## Coping with cards: Cardtester.py

- Dealing a deck of cards in Python: Cardtester.py

- In code below, what is a deck?
- What is a card?
- What's easier to understand: [0,1] or "ace of hearts"
- How do nested loops work?
- Why do we use strings? Lists? Tuples?

```
def getDeck():
    d = []
    for rank in range(0,13):
        for suit in range(0,4):
            d.append([rank,suit])
    return d
```



7.6

CompSci 6101, Spring 2012

## From Lists to Tuples

- lists, lints, lines, fines, fixes, fixed
- lists, fists, fiats, feats, seats, stats, stars, stark, stack, stuck

```
def get_roll():
    return [random.randint(1,6),random.randint(1,6)]

def track(n):
    store = set()
    for x in range(n):
        store.add(get_roll())
    return len(store)

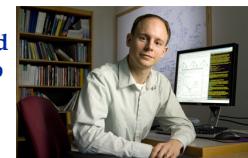
store.add(get_roll())
TypeError: unhashable type: 'list'
```

CompSci 6101, Spring 2012

7.7

## Vince Conitzer @ cs.duke.edu

Vincent Conitzer will receive the 2011 Computers and Thought Award ... The award is presented every two years to the world's leading AI researchers under the age of 35.



Conitzer, an assistant professor of computer science at Duke University, is receiving the award in recognition of his seminal work at the boundary of microeconomic theory and artificial intelligence, in particular for groundbreaking work on computational aspects of game theory, social choice, and mechanism design.

CompSci 6101, Spring 2012

7.8