Plan for FFWOO

- **Programming in the small and in the large**
  - **Making one function work correctly, especially in the context of an interactive game – *in the small***
  - **Creating multiple functions in a module that communicate with each other and "between" function calls or with other modules – *in the medium***
  - **Creating an API that other programmers can use to accomplish tasks, facilitating multi-module interactions – *toward the large***

PFFWOO continued

- Python idioms – in the small programming
  - List Comprehensions
    - `[x for x in range(100) if x % 2 == 0]`
  - Sets as simple way to structure data
    - Similar to list, but no duplicate elements

- Speaking "in the vernacular" helps in communicating with other programmers
  - Who might help you in many ways

Loops in programs

- We've seen `for X in Y:` loops
  - What type of thing has Y been?
  - What type of thing has X been?
  - Deep discussion for Y that we gloss over (iterator, iterable, and more)

- Sometimes you can't get a "next" item, but still need a loop
  - Looping to keep a program running, e.g., when interacting with user like in a game or drawing or reacting to mouse clicks or …

SmartGuessing.py

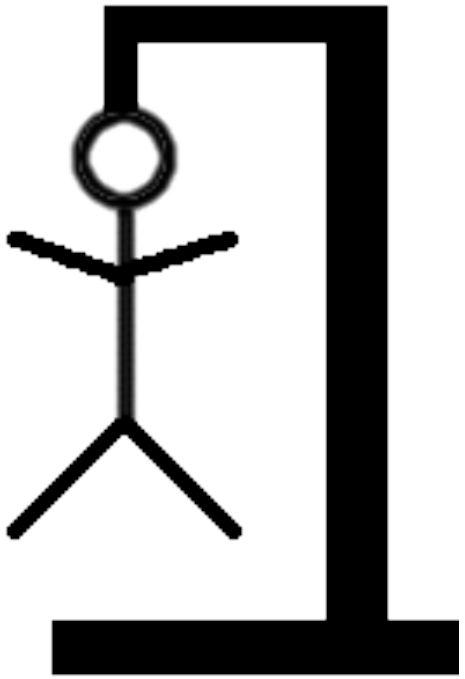
```
low, high = 1, 100
while True:
    guess = (low + high)/2
    print "I guess",guess
    response = raw_input("high/low/correct ")
    if response[0] == 'c':
        print "I guessed your number!"
        break
    elif response[0] == 'h':
        high = guess-1
    else:
        low = guess + 1
```

Thinking about guessing numbers

http://bit.ly/101fall15-1006-1

Refactoring Game-playing Program

```
while True:
    take_turn()
    update_state()
    if game_over():
        update()
        break
```



- Determine state
  - Local variables
  - Parameters to functions
  - Initialize appropriately

SmartGuessing.py

```
low, high = 1,100
while True:
    guess = (low + high)/2
    print "I guess",guess
    response = raw_input("high/low/correct ")
    if response[0] == 'c':
        print "I guessed your number!"
        break
    elif response[0] == 'h':
        high = guess-1
    else:
        low = guess + 1
```

Take Turn

Game Over

Update State

Anita Borg 1949-2003

- "Dr. Anita Borg tenaciously envisioned and set about to change the world for women and for technology. … she fought tirelessly for the development technology with positive social and human impact."
- "Anita Borg sought to revolutionize the world and the way we think about technology and its impact on our lives."
- http://www.youtube.com/watch?v=1yPxd5jqz_Q

From Numbers to Words

- **Would you like to play a game?**
  - **Words with Friends**
  - **Hanging with Friends**
  - **Jotto by yourself ☺**
- https://en.wikipedia.org/wiki/Jotto
- http://on.fb.me/1L47NSv
- http://jotto.augiehill.com/single.jsp

Problem Solving: Common APT

- `count("smart", "beast") is 3`
- `count("smart", "seats") is 3`
- `count("seems", "eases") is ?`

- General ideas:
  - We need a loop, over what?
  - We need to mark a letter as used, how?

Jotto.py

```
def play(words):
    print "Jotto: I guess your word"
    while True:
        guess = random.choice(words)
        print "my guess:",guess
        same = raw_input("how many in common? ")
        sameInt = int(same)
        if sameInt == 6:
            print "I win!!"
            break
        # conceptually what do we do here?
```

New Idiom: List Comprehension

- Given a list of strings
  - New list of just those that are "special"
  - Remove non-special strings? Create new list?

- Given a list of numbers
  - New list of just the positive numbers
  - Remove negative numbers? Create new list?

Be Positive!

```
def onlyPos(nums):
    ret = []
    for n in nums:
        if n > 0:
            ret.append(n)
    return ret

print onlyPos([1,2,3,-1,-2,-3])
```

Don't be Negative!

```
def removeNegs(nums):
    for n in nums:
        if n < 0:
            nums.remove(n)
    return nums

x = [1,2,3,-1,-2,-3]
y = removeNegs(x)
print x,y
```

List Comprehension

```
x = [1,2,-1,-2,3,4,-3,-4]
y = [n for n in x if n > 0]
```

- See onlyPos for comparison

```
def onlyPos(nums):
    ret = []
    for n in nums:
        if n > 0:
            ret.append(n)
    return ret
```

General format for list comprehension

- Creates a new list, based on existing list
- `[v_expression for v in list]`
  - v is a variable that iterates over list
  - v_expression is any expression, could use v

```
s = ['a', 'b', 'c']
t = [1,2,3]
x = [v*2 for v in s]
y = [v*2 for v in t]
```

Filtered list comprehension

- Only selects certain elements from list
- `[v_exp for v in list if bool_v]`
  - v is a variable that iterates over list
  - v_expr is any expression, could use v
  - bool_v is boolean expression, could use v

```
s = ['a', 'b', 'c']
t = [1,2,3]
x = [v*2 for v in s if v > 'a']
y = [v*2 for v in t if v % 2 == 1]
```

Questions

http://bit.ly/101fall15-1006-2

Return to Jotto

- How can we select only the words with the same number of letters in common with guess?
  - If guess is “stick” and count is 2
  - What about “stand”, “thick”, “check”
- How do we use a list comprehension?