Logistics

Homework 4
- A few Python list exercises
- Due next Thursday

Reading
- Pattern Ch 5: Algorithms And Heuristics
- Think Ch 11: Lists (link)

Mini-Exam 2
- Thursday
- Programming concepts
- Python Coding

Goals Today
- In-class Exercise
- Introduce Python lists
- Finding things in Lists
What does the next generation need to know?

- Most of you are now done with HW 3 on python programming
- We’re not done with Python but it’s a good time to reflect
- What knowledge would you bestow upon next year’s CS 100 section to make it easier for them to get through their initial difficulties with Python?
- What advice would you give a student just starting to look at Python?
- Write down your thoughts on a pieces of paper
- 1 paragraph (3-4 sentences)
- Include your name and NetID (first part of your GMU email address)
A while loop checks the condition and if it is true, continues executing the loop

```python
i = 0
while i < 10:
    print(i)
    i = i + 1
print("Done")
```

- Modify to print 0, 2, 4, 6, 8, 10, 12
- Modify to print 10, 9, 8, 7, ... 1
Consider the Following code

```
i = 1
while i > 0:
    i = i+1
print(i)
```

- What will the code print?
- How does the code flow?
- Do you see anything wrong with it?
A question is computable if a computer program can be written to answer it.

Canonical example: the Halting Problem

**Halting Problem**
Will the following computer program ever finish (halts) or will it run forever (doesn’t halt)?

*Insert your program X here.*

**Critical:** Answer for any program that could possibly be written

---

- You are all fine computers
- How would you answer the question whether a program halts or doesn’t halt?

---

**Sample Input**

X is this program below

```python
i = 100
while i > 1:
    if i % 2 == 0:
        i = i // 2
    else:
        i = i * 3 + 1
```

- "Halts" or
- "Doesn’t Halt"?
Operating Systems as Referees

- Sometimes Programs Misbehave
- The Operating System can (usually) intervene and stop the misbehaving program
- A good skill to have: know how to **kill** running programs when they misbehave
  - Windows: Task Manager
  - Mac OS X: Force Quit
  - Mobile Platforms: Varies
Farewell Turtle, Hello Text Processing

- No longer going to draw with turtle as much
- HW4 features no drawing
- Don’t need the line
  `form turtle import *`
- Instead, do calculations using text only
- Looks less exciting but often more useful and practical
Python’s Lists

- Often want to track multiple things (e.g. socks)
- Python provides a built in list data structure
- Easiest to use square braces to create them

```python
number_list = [7, 8, 6, 5, 3, 0, 9]
color_list = ["cyan", "magenta", "yellow", "black"]
```

- Can find the **length** of a list with the `len(list)` function

```python
print( len(number_list) )  # prints 7
print( len(color_list) )   # prints 4
print( len([]) )           # prints 0
```
Accessing Elements

Contents of lists are accessed by number, also uses *square braces* as in `my_list[number]`

```python
number_list = [7, 8, 6, 5, 3, 0, 9]
color_list = ["cyan", "magenta", "yellow", "black"]

print( number_list[0] ) # prints 7
print( number_list[3] ) # prints 5
print( color_list[1] ) # prints magenta
print( color_list ) # prints whole list
```
Changing Lists

Can assign specific elements like other variables

```python
number_list[0] = 8
color_list[1] = "red"
```

Can change the entire list to something else

```python
number_list = 5
color_list = ["red","green","blue"]
```
Looping and Lists

- Dealing with all contents of a list usually means looping
- Several ways to loop through the list
- Many things one can do with list loops

```python
# Make a list of numbers
numlist = [7,8,6,5,3,0,9]

# Use range to print
for i in range(len(numlist)):
    print(numlist[i])

# Loop directly to print
for number in numlist:
    print(number)

# Sum a list
total = 0
for number in numlist:
    total = total + number
print(total)

# Print even indices with if/else
for i in range(len(numlist)):
    if i%2 == 0:
        print(numlist[i])

# Print even indices with range
for i in range(0,len(numlist),+2):
    print(numlist[i])

# Print list in reverse
for i in range(len(numlist)-1,-1,-1):
    print(numlist[i])
```
Find the Biggest Number in a List

- Given a list of numbers called $L$
- All numbers are 0 or bigger
- Find the biggest number and print it
- Don’t change the list
- May be a BIG list, like 20-30 pages of numbers

Discussion: what steps are required?

List $L$ is
13453 13126 23663 4243 22574 19352 29028 26111 692 5307 14759 15903 28655 28499 17581 25411 20192 24112 12614 7826 9963 10899 10555 6347 28900 15961 470 5857 7715 10763 26601 11692 5433 1048 18304 18367 32216 11986 25581 28272 4554 32584 26081 17320 20908 805 31296 457 23454 3907 32343 9331 12084 15787 2176 20082 14587 23421 164 22112 26076 9980 14368 2118 25876 16297 17278 17362 26939 30659 25554 8621 6091 20689 5243 14391 21127 17072 25487 6140 25765 17274 12221 21804 29061 3170 9054 4225 12137 14364 24452 31329 8714 7299 9595 29015 25161 21239 8221 10002 21642 29158 2680 8019 13778 3291 19395 10797 5200 16266 17784 4783 25523 9245 18758 22210 13201 12132 7163 15486 4986 2652 21227 6965 29598 14463 28979 29916 29885 23310 5719 19711
Solution: Max Number in a List

# Find the maximum number and print it
def max_number(L):
    max = -1
    for number in L:
        if number > max:
            max = number
    print("The max is "+str(max))

Question: what would you change to find the minimum number?
Exercise: Average of Numbers

- Adapt the code for `max_number(L)` to find the average of the numbers in a list
- Call your function `list_average(L)`
- **Remember**: Exercise answers are usually distributed with the lecture slides
- Follow the *pattern* demonstrated in `max_number(L)` but will need to change some details

```python
# Find the maximum number and print it
def max_number(L):
    max = -1
    for number in L:
        if number > max:
            max = number
    print("The max is "+str(max))
```