Administrivia

- Homework due Friday
- Midterm 1 is on **Monday**!
  - Practice exam released
  - Contact cs101admin@cs.illinois.edu for conflict
<table>
<thead>
<tr>
<th>Lab Section</th>
<th>Exam Location</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>AYA AYB AYC</td>
<td>Gregory Hall</td>
<td>100</td>
</tr>
<tr>
<td>AYD AYE AYF</td>
<td>Gregory Hall</td>
<td>112</td>
</tr>
<tr>
<td>AYG AYH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AYI AYJ AYK</td>
<td>Loomis Lab</td>
<td>141</td>
</tr>
<tr>
<td>AYL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AYM AYN AYO</td>
<td>Materials Science Building</td>
<td>100</td>
</tr>
<tr>
<td>AYP AYQ AYR</td>
<td>Altgeld Hall</td>
<td>314</td>
</tr>
</tbody>
</table>
Midterm instructions

• 30 questions
• 60 minutes
• You must put in netid and exam code
  – 1 letter grade (10%) of your grade will be docked if you miss this
LISTS
Split

• *split* is a string method that returns a *list*.
• Takes a single string argument.
  – Used as a delimiter

```python
name="Ryan M. Cunningham"
m=name.split(" ")
print m[-1]
```
x = “A+B+C”
y = x.split(“+”)

What is the value of x?
a) “ABC”
b) [“A”, “B”, “C”]
c) [“+”, “+”, “+”]
d) None
Join

- A **string** method that operates on a **list**.
- Returns a **string** of list elements joined together.

```python
names=['Ryan','Dave','Michael']
','.join(names)
```
What is the value of \( x \)?

a) “XAG”
b) “X,A,G”
c) “A,G,X”
d) None
s="G R I M E S"

a=s.split(" ")

a.sort()

s=",".join(a)

print(s[ :3])

What is printed?

a) "G,R,I"
b) "G R"
c) "E,G"
d) None
s="G R I M E S"

a=s.split(" ")

b=a[:]

a.sort()

b.reverse()

x="".join(b)

What is the final value of x?

a) "GRIMES"
b) "SEMIRG"
c) "EGIMRS"
d) "SRMIGE"
TUPLES
Tuple

• A tuple is an **immutable** sequence of any type.
  – An immutable version of a list.
• Literal: item in the tuple separated by commas (can add parentheses)

```plaintext
t=(1,3.14,”Hi”)```

\begin{verbatim}
t=(1,3.14,"Hi")
t[0:2]
t[-2]
len(t)
1 in t
t[2][1]
\end{verbatim}
Why tuples?

• Less useful version of lists?
• No! They make our solutions more elegant!
• Allow us to group items together in our code.
Tuple assignment

• A tuple can go on the *left side* of an assignment statement
• Allows us to make *multiple assignments* at once

one, pi, hello = (1, 3.14, "Hi")
• Convenient for swapping values:

x, y = y, x
s="1,3,6,10"
a=s.split(",")
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?
 a) 0
    b) 8
    c) 9
    d) 11
s = "1,3,6,10"
a = s.split("","")
i = 1
x = 0

while i < len(a):
    r, s = a[i-1:i+1]
    x += int(s) - int(r)
i += 1

What is the final value of x?
a) 0
b) 8
c) 9
d) 11
s="1,3,6,10"
a=s.split("","")
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?

a) 0
b) 8
c) 9
d) 11
s="1,3,6,10"
a=s.split(“,“)
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
    i+=1

What is the final value of x?
a) 0  
b) 8  
c) 9  
d) 11
s="1,3,6,10"
a=s.split(“,“)
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?

a) 0
b) 8
c) 9
d) 11

<table>
<thead>
<tr>
<th>i_{old}</th>
<th>x_{old}</th>
<th>r</th>
<th>s</th>
<th>x_{new}</th>
<th>i_{new}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
s="1,3,6,10"
a=s.split(“,“)
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?

a) 0
b) 8
c) 9
d) 11
s="1,3,6,10"
a=s.split("","")
i=1
x=0

while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?
a) 0
b) 8
c) 9
d) 11
s="1,3,6,10"
a=s.split("","")
i=1
x=0
while i<len(a):
    r,s=a[i-1:i+1]
    x+=int(s)-int(r)
i+=1

What is the final value of x?
a) 0
b) 8
c) 9
d) 11

<table>
<thead>
<tr>
<th>i_{old}</th>
<th>x_{old}</th>
<th>r</th>
<th>s</th>
<th>x_{new}</th>
<th>i_{new}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>10</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>
Tuple return values

- A tuple can be used in a return statement
- Allows us to *return multiple values* at once

```python
def fun():
    return (1, 2, 3)
```

- When calling, can use tuple assignment

```python
a, b, c = fun()
```
String formatting with tuples

• We can use tuples on the right side of the string formatting operator.
• Allows us to insert multiple values into the string.

"%%i %%i %%i" % (1,2,3)
s=???
x=10
y="Hello"
z=3.14
print(s % x, y, z)

a) "%i %f %s"
b) "%f %s %i"
c) "%i %s %f"
d) None of the above.
Tuples and iteration

• zip - iterate through two iterables together
• Loop variable assigned a series of tuples

```python
x=[1,2,3,4]
y=“ABCD”
for a in zip(x,y):
    print a
```
Tuples and iteration

- enumerate - count as we iterate
- Loop variable contains a tuple

```
x=“ABCD”
for a in enumerate(x):
    print a
```
Exercises

1. Find all of the palindromes in words.txt
2. Find the longest palindrome in words.txt
3. Find longest word that uses only two letters.
4. Find and sort all of the even numbers in numbers.txt
def palindrome(word):
    bword=""
    for c in word:
        bword=c+bword
    return bword==word

longest_p=""
longest_l=0
for word in open("words.txt"):
    word=word.strip().lower()
    if palindrome(word):
        if len(word)>longest_l:
            longest_p=word
            longest_l=len(word)

print longest_p