CSCI 141
Computer Programming I

Filip Jagodzinski
Announcement

Homework #2 is due tonight

Homework #3 has been posted
You have until 09 February
Start right away
From Last Time

**Motivation**: Computers (via computer programs) are reaaaaly good at monotonous tasks. Humans ... no so much
From Last Time

**Motivation**: Computers (via computer programs) are reaaaaaly good at monotonous tasks. Humans … no so much

For a bank Account balance of 100.00, and assuming that each year the account earns 10% interest that is added to the account’s value, what is the account’s value after 5 years?

One “by-hand” approach is the following:

```python
balance = 100.00
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
>>>
110.0
121.0
133.1
146.41
161.051
```
**Motivation**: Computers (via computer programs) are reaaaaalyy good at monotonous tasks. Humans ... no so much

For a bank Account balance of 100.00, and assuming that each year the account earns 10% interest that is added to the account’s value, what is the account’s value after 5 years?

One “by-hand” approach is the following:

```python
balance = 100.00
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
balance = balance + (0.10 * balance)
print(balance)
```

But if we want to compute the account’s value after 500 years....
Motivation: Computers (via computer programs) are reaaaaly good at monotonous tasks. Humans ... no so much

For a bank Account balance of 100.00, and assuming that each year the account earns 10% interest that is added to the account’s value, what is the account’s value after 500 years?

```python
balance = 100.00
for x in range(0, 500):
    balance = balance + (0.10 * balance)
print(balance)
```
From Last Time

```python
for x in range(0, 3) :
    print("The value of x is", x)
```

Task: Be able to explain in your own words what the above piece of code accomplishes.
From Last Time

```python
for x in range(0, 3):
    print("The value of x is", x)
```

**Task**: Be able to explain in your own words what the above piece of code accomplishes.

The keyword `for` specifies functionality in Python for repeatedly performing a task. The iterator variable `x` takes on values output by the `range` function. For EACH value of `x`, the code block (or body) of the for loop is executed.

<table>
<thead>
<tr>
<th>x</th>
<th>The value of x is</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Today

Incrementing in for loops
Accumulators
A more detailed example
import
What values for x and y should you use so that the word “WWU” is printed out 43 times? Select all correct choices

```python
for z in range (x, y) :
    print (“WWU”)
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>A.</td>
<td>0</td>
</tr>
<tr>
<td>B.</td>
<td>1</td>
</tr>
<tr>
<td>C.</td>
<td>-21</td>
</tr>
<tr>
<td>D.</td>
<td>-21</td>
</tr>
<tr>
<td>E.</td>
<td>-789</td>
</tr>
<tr>
<td>F.</td>
<td>-789</td>
</tr>
</tbody>
</table>
What values for \( x \) and \( y \) should you use so that the word “WWU” is printed out 43 times? Select all correct choices:

```python
for z in range (x, y) :
    print ("WWU")
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>0</td>
</tr>
<tr>
<td>B.</td>
<td>1</td>
</tr>
<tr>
<td>C.</td>
<td>-21</td>
</tr>
<tr>
<td>D.</td>
<td>-21</td>
</tr>
<tr>
<td>E.</td>
<td>-789</td>
</tr>
<tr>
<td>F.</td>
<td>-789</td>
</tr>
</tbody>
</table>

As a first step, what is the smallest and largest number that \( \text{range} \) generates for these inputs of \( x \) and \( y \), and then you must answer how many integers numbers are “in between” that minimum and maximum.
What values for $x$ and $y$ should you use so that the word “WWU” is printed out 43 times? Select all correct choices.

```
for z in range (x, y) :
    print (“WWU”)
```

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>0</td>
<td>44</td>
</tr>
<tr>
<td>B.</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>C.</td>
<td>-21</td>
<td>21</td>
</tr>
<tr>
<td>D.</td>
<td>-21</td>
<td>22</td>
</tr>
<tr>
<td>E.</td>
<td>-789</td>
<td>746</td>
</tr>
<tr>
<td>F.</td>
<td>-789</td>
<td>-746</td>
</tr>
</tbody>
</table>

Recall that range (when it is given just two parameters) generates a list of numbers in increments of 1 starting from $x$ and up to but not including $y$. 

- A. $0, 1, 2, 3, \ldots, 41, 42, 43$
- B. $1, 2, 3, 4, \ldots, 42, 42, 43$
- C. $-21, -20, -19, \ldots, 20, 21$
- D. $-21, -20, -19, \ldots, 21, 22$
- E. $-789, -788, \ldots, 744, 745$
- F. $-789, -788, \ldots, 747, 746$
Warm-up

What values for \( x \) and \( y \) should you use so that the word “WWU” is printed out 43 times? Select all correct choices.

```python
for z in range(x, y):
    print("WWU")
```

| A. | 0   | 44   | 0, 1, 2, 3, …, 41, 42, 43 |
| B. | 1   | 44   | 1, 2, 3, 4, …, 42, 43    |
| C. | -21 | 21   | -21, -20, -19, …, 19, 20 |
| D. | -21 | 22   | -21, -20, -19, …, 20, 21 |
| E. | -789| 746  | -789, -788, …, 744, 745  |
| F. | -789| -746 | -789, -788, …, -747, -745 |

Recall that `range` (when it is given just two parameters) generates a list of numbers in increments of 1 starting from \( x \) and up to but not including \( y \).

How many integers are in these ranges?
Warm-up

What values for x and y should you use so that the word “WWU” is printed out 43 times? Select all correct choices

```
for z in range (x, y) :
    print ("WWU")
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>Choice</th>
<th>Notes</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>44</td>
<td>A.</td>
<td>0, 1, 2, 3, ... 41, 42, 43</td>
<td>44</td>
</tr>
<tr>
<td>1</td>
<td>44</td>
<td>B.</td>
<td>1, 2, 3, 4, ..., 42, 42, 43</td>
<td>43</td>
</tr>
<tr>
<td>-21</td>
<td>21</td>
<td>C.</td>
<td>-21, -20, -19, ..., 19, 20</td>
<td>42</td>
</tr>
<tr>
<td>-21</td>
<td>22</td>
<td>D.</td>
<td>-21, -20, -19, ..., 20, 21</td>
<td>43</td>
</tr>
<tr>
<td>-789</td>
<td>746</td>
<td>E.</td>
<td>-789, -788, ..., 744, 745</td>
<td>1535</td>
</tr>
<tr>
<td>-789</td>
<td>-746</td>
<td>F.</td>
<td>-789, -788, ..., -747, -745</td>
<td>43</td>
</tr>
</tbody>
</table>

Recall that range (when it is given just two parameters) generates a list of numbers in increments of 1 starting from x and up to but not including y.
What values for x and y should you use so that the word “WWU” is printed out 43 times? Select all correct choices.

```
for z in range (x, y) :
    print ("WWU")
```

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>0</td>
<td>44</td>
</tr>
<tr>
<td>B.</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>C.</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>D.</td>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>E.</td>
<td>-789</td>
<td>746</td>
</tr>
<tr>
<td>F.</td>
<td>-789</td>
<td>-746</td>
</tr>
</tbody>
</table>

Recall that range (when it is given just two parameters) generates a list of numbers in increments of 1 starting from x and up to but not including y.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range(0, 100):
    print(x)
```

Would this work? Why or why not?
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range(0, 100):
    print(x)
```

Would this work? Why or why not?

No. This prints out ALL numbers from 0 through 99.

However you’ve learned of an operator that you can use in combination with the equality operator to see if a number divisible by another number gives a zero remainder.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

Step-by-step walk through
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

This will generate a list of numbers from 0 up to but not including 100

0, 1, 2, 3, 4, ..., 99
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

Notice the indentation:

This is the code block (or body) of the for loop.

Q: When is it executed?
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

Notice the indentation:

This is the code block for the if statement.

Q: When is this code block executed?
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

This will generate a list of numbers from 0 up to but not including 100:

0, 1, 2, 3, 4, ..., 99

x is the loop (or iterator) variable, so for each value of x, the code block (or body) of the for loop will be executed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

Q: What is 0 modulus 2?

This will generate a list of numbers from 0 up to but not including 100:

\(0, 1, 2, 3, 4, ..., 99\)

When \(x=0\), the code block will be executed. What does the code block do? It checks if 0 modulus 2 is equal to 0. If it is, then the value of \(x\) is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

Q: What is 0 modulus 2?

\[ 2 \div 0 \]

This will generate a list of numbers from 0 up to but not including 100

0, 1, 2, 3, 4, ..., 99

When \( x=0 \), the code block will be executed. What does the code block do? It checks if 0 modulus 2 is equal to 0. If it is, then the value of \( x \) is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

Q: What is 0 modulus 2?

$$
\begin{array}{c}
0 \\
2) 0 \\
0 \\
0
\end{array}
$$

This will generate a list of numbers from 0 up to but not including 100:

$$0, 1, 2, 3, 4, ..., 99$$

When x=0, the code block will be executed. What does the code block do? It checks if 0 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100

Q: What is 0 modulus 2?

```
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

This will generate a list of numbers from 0 up to but not including 100:

```
0, 1, 2, 3, 4, ..., 99
```

When x=0, the code block will be executed. What does the code block do? It checks if 0 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

This will generate a list of numbers from 0 up to but not including 100

Output

0

0, 1, 2, 3, 4, ..., 99

When x=0, the code block will be executed. What does the code block do? It checks if 0 modulus 2 is equal to 0. If it is, then the value of x is printed
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

This will generate a list of numbers from 0 up to but not including 100

Output

0

0, 1, 2, 3, 4, ..., 99

When x=1, the code block will be executed. What does the code block do? It checks if 1 modulus 2 is equal to 0. If it is, then the value of x is printed
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

Output
0
2

This will generate a list of numbers from 0 up to but not including 100:
0, 1, 2, 3, 4, ..., 99

When x=2, the code block will be executed. What does the code block do? It checks if 2 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

Output
0
2

```python
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

This will generate a list of numbers from 0 up to but not including 100

0, 1, 2, 3, 4, ..., 99

When x=3, the code block will be executed. What does the code block do? It checks if 3 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

Output
- 0
- 2
- 4

This will generate a list of numbers from 0 up to but not including 100:
- 0, 1, 2, 3, 4, ..., 99

When x = 4, the code block will be executed. What does the code block do? It checks if 4 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

Output

0
2
4
...
98

This will generate a list of numbers from 0 up to but not including 100

0, 1, 2, 3, 4, ..., 99

When x=99, the code block will be executed. What does the code block do? It checks if 99 modulus 2 is equal to 0. If it is, then the value of x is printed.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

Q: Is there another way to accomplish this same task?

Q: Can this be done with 2 lines of code?

Output:

0
2
4
...
98
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

Q: Is there another way to accomplish this same task?

The range function can have a third argument, that specifies the increment value of each successive number.

```python
for x in range (0, 100, 2):
    print (x)
```
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write Python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range(0, 100):
    if (x % 2 == 0):
        print(x)
```

Q: Is there another way to accomplish this same task?

The range function can have a third argument, that specifies the increment value of each successive number.

```python
for x in range(0, 100, 2):
    print(x)
```

This specifies “generate” numbers from 0 up to but not including 100, in increments of 2.
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range (0, 100):
    if (x % 2 == 0) :
        print (x)
```

Q: Do these pieces of code output the same thing?

```python
for x in range (0, 100, 2):
    print(x)
```
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100

```python
for x in range (0, 100):
    if (x % 2 == 0):
        print(x)
```

Q: Do these pieces of code output the same thing?

They do.

Q: Is one better than the other?
Other “forms” of for-loops

Assume that you want to print out all numbers from 0 to 100 that are even. How could you do this using Python and what you’ve learned already?

Task: Write python code that uses a for loop to print out all even numbers from 0 up to and not including 100.

```python
for x in range (0, 100):
    if (x % 2 == 0):
        print (x)
```

Q: Do these pieces of code output the same thing?

They do.

Q: Is one better than the other?

Not necessarily.
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    aValue = aNum
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    aValue = aNum
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

As a first step, what list of numbers does `range(4, 6, -2)` generate?
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    aValue = aNum
    print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

As a first step, what list of numbers does `range(4, 6, -2)` generate?

**No list.** Starting at 4, proceeding in increments of -2 “until” it reaches 6, does not generate a list because there is NO such sequence of numbers
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    aValue = aNum
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

4, 6, -2  →  range  →  {}
Q: What does the following code output?

```
for aNum in range (4, 6, -2) :
    aValue = aNum
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

```
4,6,-2  range  \{\}
```

\text{aNum} \text{ assumes, in turn, each of the values in the list generated by range, but range doesn’t return a list, therefore how many times does the code block for the for loop execute?}
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    # never executed
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

Zero times. Therefore, the variable aValue is never declared nor assigned a value, so when the print statement attempts to “use” the value for the variable aValue ...
For loops ... another “tricky” example

Q: What does the following code output?

```python
for aNum in range (4, 6, -2) :
    never executed
print ("The value of aValue is", aValue)
```

A. The value of aValue is 4
   The value of aValue is 6
B. The value of aValue is 6
   The value of aValue is 4
C. nothing
D. An error message because the code has an error

An error occurs. Specifically, a Name Error, which occurs when you try to use a variable before it has been assigned a value
Remember that the code block (or body) of a for loop can contain multiple lines of code.

```python
interest = 0.0
accountBalance = 100.00
for x in range (1, 6) :
    interest = accountBalance * 0.10
    accountBalance = accountBalance + interest
print ("In year x, interest is ", interest,
    "and balance is", accountBalance)
```

Q: What does this code accomplish
Accumulators

Remember that the code block (or body) of a for loop can contain multiple lines of code.

```python
interest = 0.0
accountBalance = 100.00
for x in range (1, 6):
    interest = accountBalance * 0.10
    accountBalance = accountBalance + interest
print ("In year x, interest is ", interest,
      "and balance is", accountBalance)
```

Identify the code block
Q: How many iterations are performed?
Q: What is the output?

(Step through on the board)
Accumulators

Remember that the code block (or body) of a for loop can contain multiple lines of code.

```python
interest = 0.0
accountBalance = 100.00
for x in range (1, 6):
    interest = accountBalance * 0.10
    accountBalance = accountBalance + interest
print ("In year x, interest is ", interest,
       "and balance is", accountBalance)
```

These are often referred to as accumulators because they are updated (or accumulate values) in the body of the for loop.
Q: What is the output of the following code?

```python
aValue = 0
for aNum in range (7, 9) :
    aValue = aNum + aNum
print(aValue)
```
Q: What is the output of the following code?

```python
aValue = 0
for aNum in range (7, 9):
    aValue = aNum + aNum
print(aValue)
```

on the board explanation
Import

As you’ve learned (will learn if you haven’t yet started it) from lab 4, there is existing code, or functionality, available for your use which is part of the core of python. Lab 4 teaches you how to generate random numbers, and that capability is in the `random` module.

To use the capability in a module, you must explicitly `import` it

```
import random
```
As you’ve learned (will learn if you haven’t yet started it) from lab 4, there is existing code, or functionality, available for your use which is part of the core of python. Lab 4 teaches you how to generate random numbers, and that capability is in the `random` module.

To use the capability in a module, you must explicitly import it

```python
import random
```

Q: What functionality and/or capabilities are available for your use in the `random` module?
As you’ve learned (will learn if you haven’t yet started it) from lab 4, there is existing code, or functionality, available for your use which is part of the core of python. Lab 4 teaches you how to generate random numbers, and that capability is in the random module.

To use the capability in a module, you must explicitly import it

```python
import random
```

Q: What functionality and/or capabilities are available for your use in the random module?

Just as existing functions in python contain code that somebody has written and you should think of as a black box, the random module, too, has code that somebody has written and which is now “part” of python. You can see the code if you’d like ...

https://hg.python.org/cpython/file/3.4/Lib/random.py (demo)
As you’ve learned (will learn if you haven’t yet started it) from lab 4, there is existing code, or functionality, available for your use which is part of the core of python. Lab 4 teaches you how to generate random numbers, and that capability is in the `random` module.

To use the capability in a module, you must explicitly `import` it.

**Q: What functionality and/or capabilities are available for your use in the `random` module?**

Don’t worry about the “code” that is inside the `random` module.

- You don’t need to memorize it
- If you “read” it you won’t understand much of what is there
- You’ll learn about classes, the `self` keyword, class variables, overriding, inheritance, overloading, class functions, etc. in later CS courses

For the time being just be thankful that somebody has written these modules, which allows you to write fairly complicated programs.
Task: Write a program that prints 11 random numbers

Q: How do you “start”
Import

Task: Write a program that prints 11 random numbers

Q: How do you “start”

Looking at the API, there are many functions available for your use when you import the `random` module. They include:

```python
random()
getrandbits(k)
sample(population, k)
randint(a, b)
randrange(start, stop, step)
randrange(start, stop)
```
Task: Write a program that prints 11 random numbers

Q: How do you “start”

Looking at the API, there are many functions available for your use when you import the `random` module. They include:

- `random()`
- `getrandbits(k)`
- `sample(population, k)`
- `randint(a, b)`
- `randrange(start, stop, step)`
- `randrange(start, stop)`

Assume you select this one...

You’ll gain experience with many of these. Which one you select to use is dependent on the problem that you want to solve.
Task: Write a program that prints 11 random numbers

```python
aVar = random()
print(aVar)
```

Q: Does this work? Why or why not?
Task: Write a program that prints 11 random numbers

```python
aVar = random()
print(aVar)
```

Q: Does this work? Why or why not?

This generates an error because in order to use the functions in a module (that is not part of the core of python) you must first import that module

Q: What is the fix?
Task: Write a program that prints 11 random numbers

```python
import random
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
```

Q: What does this code output?
Task: Write a program that prints 11 random numbers

```
import random
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())
print(random.random())

Q: What does this code output?
```

This is OKAY, but what if you want integers!
Import

Task: Write a program that prints 11 random numbers (in class demo)

Back to the API ...

Write two programs, which use DIFFERENT functions from random, and which print to the screen 11 random integers
Take home exercise

Write python code that will print the first 10 positive multiples of the number 34 whose ones digit is NOT 0 nor 5.

Hint: the digits are 34, 68, 102, 136, 204, 238, 272, 306, 374, 408
Up next

Lists
Functions