CSC 111: Intro to Computer Science through Programming

Spring 2017
Prof. Sara Mathieson
Homework 6 is due March 28
No transcripts for graphics (screenshots instead)
I will be away next week at a conference, but there will still be class, lab, and homework as usual (but no office hours)
Outline: 3/24

- Recap last time
- Midterm feedback
- Continue: animated graphics
- User input in graphics
Recap
(+ snow examples)
Chelsey
Tani
import random

def main():
    # ask the user for a random number
    number = eval(input("Enter a number: "))

    # start off with an initial guess
    guess = random.randint(1,100)
    print(guess)

    # while the guess is not equal to the user's number, guess again
    while guess != number:
        guess = random.randint(1,100)
        print(guess)

    # after the while loop is over, the computer must
    # have guessed the number!
    print("You guessed it!")

main()
Midterm Feedback
Part 1

(a) What numbers are printed when the following code is executed?

(b) What numbers are printed when the following code is executed?
Part 1

(a) What numbers are printed when the following code is executed?

```python
>>> for i in range(6,-8,-2):
    print(i)

6
4
2
0
-2
-4
-6
```

(b) What numbers are printed when the following code is executed?

```python
>>> for i in range(-3,-1):
    print(i)

-3
-2
```
(c) Describe the issue with the following code, which is attempting to return the sum of the numbers in a list.

```python
def my_sum(lst):
    s = 0
    for i in lst:
        s = s + lst[i]
    return s
```
(c) Describe the issue with the following code, which is attempting to return the sum of the numbers in a list.

```python
def my_sum(lst):
    s = 0
    for i in lst:
        s = s + lst[i]
    return s
```

- `i` is an element of the list, not an index, so trying to index in this way will produce errors (changing `i` to `elem` or something similar would help).

```
s = s + lst[i]     # Error
IndexError: list index out of range
```

```
s = s + lst[i]
TypeError: list indices must be integers, not float
```
(d) If I invoke the main method below, what will be the value of \( x \)?

```python
def compute(a, b, c):
    return a*(b-c)

def main():
    a = 1
    b = 3
    c = 2
    x = compute(b, c, a)
```
Part 1

(d) If I invoke the main method below, what will be the value of x?

```python
def compute(a, b, c):
    return a*(b-c)

def main():
    a = 1
    b = 3
    c = 2
    x = compute(b, c, a)
    >>>> compute(3, 2, 1)
    >>>> 3*(2-1)
    x = 3
```

These a, b, c variables are completely separate from these a, b, c variables.
(e) What is wrong with this attempt to read a file? How could you modify the variable names to help prevent this type of error?

```python
words = open("my_file.txt","r")
for text in words:
    x = words.count("a")
    text
```

I would change:
```
\{ words \to file
    text \to line
```

- The main issue is that `words` is the file object, so we cannot call `words.count("a")` should be `text.count("a")`.
- We should also close the file!
(f) The following lines of code are executed sequentially in the shell. In the right column, write out what is printed after each line (some lines of code may not have any output). If a line produces an error, just write “error”. Then circle all the lines that make use of casting.
(f) The following lines of code are executed sequentially in the shell. In the right column, write out what is printed after each line (some lines of code may not have any output). If a line produces an error, just write “error”. Then circle all the lines that make use of casting.

<table>
<thead>
<tr>
<th>line of code</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt; x = &quot;3.14&quot;</td>
<td></td>
</tr>
<tr>
<td>&gt;&gt;&gt; float(x)</td>
<td>3.14</td>
</tr>
<tr>
<td>&gt;&gt;&gt; x = x + 1</td>
<td>error</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y = float(x)</td>
<td></td>
</tr>
<tr>
<td>&gt;&gt;&gt; y</td>
<td>3.14</td>
</tr>
<tr>
<td>&gt;&gt;&gt; z = int(y)</td>
<td></td>
</tr>
<tr>
<td>&gt;&gt;&gt; z</td>
<td>3</td>
</tr>
<tr>
<td>&gt;&gt;&gt; type(z)</td>
<td>&lt;class 'int'&gt;</td>
</tr>
<tr>
<td>&gt;&gt;&gt; type(y)</td>
<td>&lt;class 'float'&gt;</td>
</tr>
<tr>
<td>&gt;&gt;&gt; type(x)</td>
<td>&lt;class 'str'&gt;</td>
</tr>
</tbody>
</table>
(g) Write a function in the box below that will take one parameter, a list of full names, and modify the list so that it only contains the first names. A few examples are shown below:

```python
>>> lst1 = ["Katherine Johnson","Dorothy Vaughan","Mary Jackson"]
>>> first_name(lst1)
>>> lst1
[‘Katherine’, ‘Dorothy’, ‘Mary’]
>>> lst2 = ["Logan Swanson","Artemis Metaxa","Val McCulloch","Zoe Kendall"]
>>> first_name(lst2)
>>> lst2
```
Part 1

(g) Write a function in the box below that will take one parameter, a list of full names, and modify the list so that it only contains the first names. A few examples are shown below:

```python
>>> lst1 = ['Katherine Johnson', 'Dorothy Vaughan', 'Mary Jackson']
>>> first_name(lst1)
>>> lst1
['Katherine', 'Dorothy', 'Mary']
>>> lst2 = ['Logan Swanson', 'Artemis Metaxa', 'Val McCulloch', 'Zoe Kendall']
>>> first_name(lst2)
>>> lst2
['Logan', 'Artemis', 'Val', 'Zoe']
```

```python
def first_name(lst):
    for i in range(len(lst)):
        first_last = lst[i].split()
        lst[i] = first_last[0]
```
Part 2

The goal for this question is to write a function `capital_index(string)`, which returns the index of the first capital letter in a string. If there are no capital letters in the string, the function should print "No capital letters found". Here are a few examples of this function in the shell:

```python
>>> capital_index("Spring")
0
>>> capital_index("brEaK")
2
>>> capital_index("hello")
No capital letters found
>>> capital_index("computeR")
7
>>> capital_index("sciEnCE")
3
```
The goal for this question is to write a function `capital_index(string)`, which returns the *index* of the first capital letter in a string. If there are no capital letters in the string, the function should print "No capital letters found". Here are a few examples of this function in the shell:

```python
>>> capital_index("Spring")
0
>>> capital_index("brEaK")
2
>>> capital_index("hello")
No capital letters found
>>> capital_index("computer")
7
>>> capital_index("science")
3
```

```python
# returns whether or not a single letter is a capital
# example: "a" will return False and "A" will return True
def is_capital(letter):
    if letter == letter.upper():
        return True
    return False

def capital_index(string):
    for i in range(len(string)):
        if is_capital(string[i]):
            return i
    print("No capital letters found")
```
import random

def mystery_method1(n):
    r_lst = []
    for i in range(n):
        r_lst.append(random.randint(0, 19))
    return r_lst

def mystery_method2(lst):
    for j in range(len(lst)-1):
        if lst[j] > lst[j+1]:
            return False
    return True

def main():
    num = eval(input("Enter a number: "))
    my_lst = mystery_method1(num)
    result = mystery_method2(my_lst)

    print(my_lst)
    print(result)

main()
Part 3

```python
import random

def mystery_method1(n):
    r_lst = []
    for i in range(n):
        r_lst.append(random.randint(0, 19))
    return r_lst

def mystery_method2(lst):
    for j in range(len(lst) - 1):
        if lst[j] > lst[j+1]:
            return False
    return True

def main():
    num = eval(input("Enter a number: "))
    my_lst = mystery_method1(num)
    result = mystery_method2(my_lst)

    print(my_lst)
    print(result)

main()
```

Formal parameters:
- n, lst

Actual parameters:
- num, my_lst

randint:
- function

Goal: find out if the list is in ascending order (i.e. sorted low to high)
The following examples show an inverted triangle being printed after the user enters an integer. Write and call a main method to produce this result. Notice that if the number is 3, there are 7 stars in the first row, and if the number is 5, there are 11 stars in the first row. Your main method should be general for any integer.
The following examples show an inverted triangle being printed after the user enters an integer. Write and call a main method to produce this result. Notice that if the number is 3, there are 7 stars in the first row, and if the number is 5, there are 11 stars in the first row. Your main method should be general for any integer.

```python
def main():
    n = eval(input("Enter a number: "))
    print("*"*(2*n+1))
    for i in range(1,n):
        print(" "*i + "*" + " "*(2*(n-i)-1) + "*")
    print(" "*n + "*")
main()
```
In class we have seen how to swap the values of two variables, and in homework we have seen how to swap the values of two elements in a list. In this question, the goal is to swap the values of three variables, so that each variable ends up with the value of the “previous” variable. For example, if \( x = 6 \), \( y = 3 \), and \( z = 1 \), then the end result should be \( x = 1 \), \( y = 6 \), and \( z = 3 \).

(a) The following code shows a first attempt at this process. Fill in the table below, showing what will happen after each line is executed. The first row has been filled in with the initial values from the example above.
Part 5

In class we have seen how to swap the values of two variables, and in homework we have seen how to swap the values of two elements in a list. In this question, the goal is to swap the values of three variables, so that each variable ends up with the value of the “previous” variable. For example, if $x = 6$, $y = 3$, and $z = 1$, then the end result should be $x = 1$, $y = 6$, and $z = 3$.

(a) The following code shows a first attempt at this process. Fill in the table below, showing what will happen after each line is executed. The first row has been filled in with the initial values from the example above.

<table>
<thead>
<tr>
<th></th>
<th>code</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>temp1</th>
<th>temp2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>temp1 = x</td>
<td></td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>temp2 = y</td>
<td></td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>x = z</td>
<td></td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>y = x</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>z = y</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>
(d) Would it be possible to write a function to swap three variables? i.e. would it be possible to write a function that produces the output below? Explain your answer and reasoning, using the concept of mutable vs. immutable types.

```python
>>> x = 6
>>> y = 3
>>> z = 1
>>> three_way_swap(x, y, z)
>>> x
1
>>> y
6
>>> z
3
```
(e) Write a function that will return a new string with the $i^{th}$ character and $j^{th}$ character swapped. Example: `string_swap("spring",1,5)` should return "sgrinp". You may assume that $i$ is less than $j$. Why must this function return a new string?

```python
def string_swap(string, i, j):
```
Notes about grades

• This exam was very challenging!
• Think about the midterm not as a number, but as an opportunity for improvement
• Trajectory is very important
• Midterm is only worth 15%
• Grade on Moodle is lower bound for final_hw/lab grade
Midterm curve

- 95-100: A+
- 90-94: A
- 85-89: A-
- 80-84: B+
- 70-79: B
- 65-69: B-
- 60-64: C+
- 55-59: C
- 50-54: C-
- < 50: not passing

Average: 77
Continue: animated graphics