Lists and Loops Unit

by Luis Collado

General Overview

(include here description of unit, what class(es) it fits into, when...)
In my theoretical class of Intro to Programming with Python with Turtle, we'll have a unit called lists and loops. We'll start with `while`, then learn `for`, and then learn lists. This unit would go inside of the fictional class "Intro to Python featuring Turtle", because Turtle isn't used all the time in the class. Before this class, students would have already mastered:

- **Unit 1** - Variables & Python
- **Unit 2** - Conditionals

Students would then go on to learn functions, classes, recursion, etc. This unit wouldn't be super far in the year, but isn't the first thing students are learning.

This fictional class is highly structured - every topic gets its practice time, and every new topic means it's time for a quiz on the previous topic.

Motivation for Unit

In an introductory course, you need to cover control structures and data structures. I think that the list is the most important data structure in Python, so it should be covered before tuples and dictionaries. My reason for writing this unit with Turtle is that Python with Turtle is a useful tool for instruction - if it helps, then it is available, and if it doesn't help students, then there is the rest of the unit work to support their learning of a topic. Sometimes what can make something click for a student is an alternate mode of presentation of the same content.

Standards Referenced

List the NY State Computer Science standards you will be covering in this unit. Provide the number and name (e.g. 4-6.CT.1 Computational Thinking, Modeling and Simulation)

9-12.CT.4 - Implement a program using a combination of student-defined and third-party functions to organize the computation.

9-12.CT.8 - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.

9-12.CT.9 - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.

9-12.CT.10 - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Tools Used

(include programming language(s), specific programs/environments, and other tools (digital or otherwise) if necessary)

- Code.org's The Artist
- Replit
- Pen and paper

Resources

(include any links/books/readings to be used during this unit)

- Turtle Cheat Sheet

Lessons

Total length: 2 Weeks

(list each lesson with main topic(s))

Class 0 - Conditionals Review & Intro to while loops

Class 1 - While loops lab 1

Class 2 - While loops lab 2

Class 3 - While loops quiz, Intro to for loops

Class 4 - For loops lab 1

Class 5 - For loops lab 2

Class 6 - for loops quiz, intro to lists

Class 7 - while loops and lists lab

Class 8 - for loops and lists lab

Class 9 - lists quiz, final lab

Class 10 - final lab + review time

Class 11 - lists + loops test

Assesments

- **Summative Assessments**
  - three summative assessments take place in the form of re-takable review quizzes.
  - one final summative assessment is administered at the end of the unit.
- **Formative Assessments**
  - Students are consistently producing work, whether it is the practice exercises, lab assignments, or extension work. This work is all collected and used to inform instruction.

Class 00 - intro to while loops

Learning Objective[s]

- Students will be able to explain in words what a while loop is and how it is similar to an if statement in Python.
- Students will be able to write simple while loops independently.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

```
"""
Review: Conditionals are used after 'if' statements.

They check if something is true or if something equals something.

You can use:
- and
- not
- or
- ()
- ==
- !=

examples:

name = "Luis"
```

```
if name != "Luis":
  print("Hello.")
else:
  print("Hello, Luis.")

Exercise 0: Modify the following code so that the sentences printed out are
true.
"""

age = 3

if age < 4: # change this line
  print("People aged", age, "and higher are allowed to drive.")


if age < 4: # change this line
  print("People aged", age, "and higher are allowed to buy cigarettes.")


if age < 4: # change this line
  print("People that are", age, "years old are considered teenagers.")


"""
Indefinite Loops

Indefinite Loops are like if statements, except that the conditional at the
top is RE-RUN AT THE END OF THE BLOCK.

Exercise 1:
Below is an infinitely running while loop.
Fix it to print the numbers 0-4, inclusive.
"""

# # Uncomment these lines by highlighting all of them and pressing Ctrl + /

# print("Exercise 1:")
# count = 0

# while count == 0:
#   print(count)

"""
Exercise 2:

Write a while loop that prints out the numbers 0-10, inclusive.
"""
```

```
# put exercise 2 here.

# print("Exercise 2:")


"""
Exercise 3:

Write a while loop that prints the following output:
a
aa
aaa
aaaa
aaaaa
aaaaaa
"""
# write exercise 3 here.

# print("Exercise 3:")

"""
Exercise 4:

Write a while loop that prints every even number from 0 to 100.
"""

# write exercise 4 here.

# print("Exercise 4:")

"""
Exercise 5:
Write a while loop that prints the first 20 powers of 2.
(1, 2, 4, 8, 16, 32, 64, 128, etc...)
"""

# write exercise 5 here.

# print("Exercise 5:")

"""
Exercise 6:
Write a short program that takes a number from the user, and counts down
from that number.
"""
```

```
# write exercise 6 here.

# print("Exercise 6:")

# number = int(input("What number are we counting down from?"))


"""
Independent practice:

Using the modulo operator (%) and while and if,
write a program that prints out all of the
factors of a number. Here is a useful (!) snippet
of code that prints a factor if it is a factor of
a number.

if number % factor == 0:
  print(factor)

Your program should:
1. Ask the user for a number.
2. Create a variable called factor
3. Have a while loop that uses factor and number in the conditional
4. Have an if statement in the while loop
5. Have a print statement in the if block
"""

# write your independent practice program below.



"""
Optional challenges:

1. Write a program that asks the user for a number, n,
and prints the first n numbers of the Fibonacci sequence.

2. Write a program that makes a zigzag pattern
in the terminal, like this:
_____
____
___
__
_

_
__
```

```
___
____
_____
____
___
etc.
Hint: You will need 3 while loops for this.
"""

# write your optional challenge programs below.
```

In-Class Exercises

Do Now: write a conditional the prints your name if you're old enough to drive.

The following demo code will be used:

```
a = 0
if a < 1:
  print("a is less than 1!")

while a < 1:
  print("a is less than 1")


while a < 1:
  print(a)


while a < 1:
  print(a)
  a = a + 1 # a += 1

a = 10

while a < 1:
  print("a is less than 1")


while a > 0:
  print("a is more than 10")
  a = a - 1
```

The 6 exercises listed above are the in-class activity for the day. Before any coding is done, while loops are demonstrated on the board.

Assignments

On Google Classroom:

Problems 5.1.1, 5.1.2, and 5.1.3 from: https://runestone.academy/ns/books/published/CS1-Python-Subgoals/while_counter.html?mode=browsing

Write 1 sentence explaining your answer as a private comment on google classroom.

Class 01 - indefinite practice I

Learning Objective[s]

- Students will be able to use their knowledge of conditionals to support their practice with while loops.
- Students will be able to make a playable number guessing game in Python.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

Demo with extension at end:

```
Welcome to the guessing game!
Enter a guess: 3
Too high!
Enter a guess: 2
Congratulations! You got it!
It took you 2 tries.
You set a new high score!
Type your name: Luis
Type y and press enter to play again.y
Enter a guess: 2
Too low!
Enter a guess: 3
Too low!
Enter a guess: 4
Too low!
Enter a guess: 5
Congratulations! You got it!
It took you 4 tries.
The current high score is 2 and was set by Luis .
Type y and press enter to play again.y
Enter a guess: 3
Too low!
Enter a guess: 4
Congratulations! You got it!
It took you 2 tries.
The current high score is 2 and was set by Luis .
Type y and press enter to play again.y
Enter a guess: 5
Too high!
Enter a guess: 4
Too high!
Enter a guess: 3
Too high!
Enter a guess: 2
Congratulations! You got it!
It took you 4 tries.
The current high score is 2 and was set by Luis .
Type y and press enter to play again.y
```

```
"""
Indefinite (while) Loops Lab I

In this lab, we will make a guessing game using while loops.

The computer will generate a random number between 0 and 100.

The user will guess what the number is.

After each guess, the computer will say, "Too low!"
or "Too high!" to guide the user to the correct answer.

When the user finally gets the answer right, the computer
will tell them that they are correct and tell them how many
tries it took them to get the number correct.

Example game:
Welcome to the number guessing game!
Guess a number: 5
Too low!
Guess again: 28
```

```
Too low!
Guess again: 57
Too high!
Guess again: 44
Too low!
Guess again: 50
Too low!
Guess again: 54
Too high!
Guess again: 51
Congratulations, you got it!
It only took you 7 tries.

Extensions to lab 1:
1. Expand your program to allow the user to play again
without re-running the program.
2. Expand your program to keep track of a high score.
Whenever the user sets a high score,
have the program ask their name.
"""
```

In-Class Exercises

Do Now: think of how you could implement a password login using while loops in python.

The above file is the in-class activity and extension work for the day.

Assignments

5.2.1, 5.2.2, 5.2.3, 5.2.4, and 5.2.5 from https://runestone.academy/ns/books/published/CS1-Python-Subgoals/loops-whc-p1.html?mode=browsing

Write 1 sentence for each multiple choice question as a private comment on the Google Classroom assignment.

Class 02 - indefinite practice ii

Learning Objectives

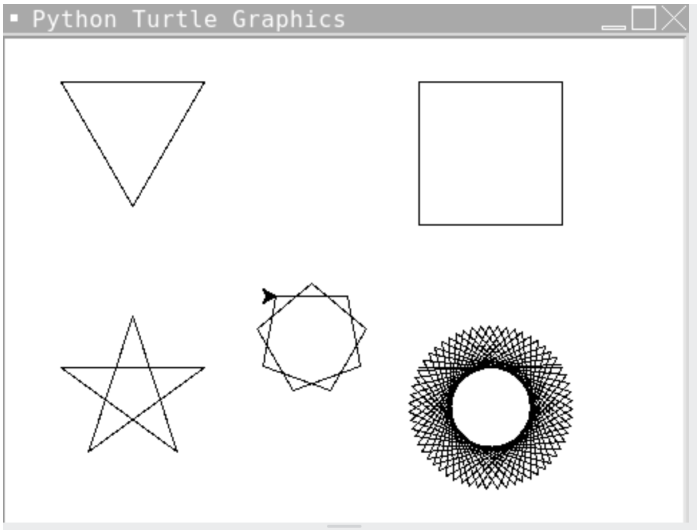
- Students will be able to apply their knowledge of while loops to create designs that meet specific criteria in Python with Turtle.
- Students will be able to work with starter-code constraints to construct their own program.

Standards

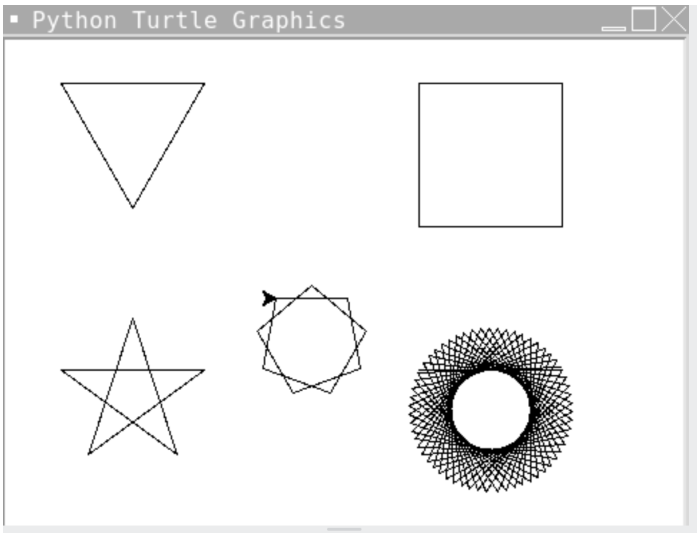
- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.

Materials & Resources

`while` lab 2 example output



`while` lab 2 extension example output



Students will be coding using this file as starter code:

```
"""
Indefinite Loops (while) Lab II

Use this starter code to create the design goal.

Don't change the setup code OR the code sandwiched between comments that
say "DON'T TOUCH!"

"""

# setup - do not change!
import turtle

t = turtle.Turtle()
s = turtle.Screen()
s.setup(1.0, 1.0)
t.penup()
t.speed(0)

t.goto(-200, 150)
t.pendown()
t.setheading(0)



# BEGIN TRIANGLE CODE --- vvvvvvvvvvvvvvvvv


# END TRIANGLE CODE --- ^^^^^^^^^^^^^^^^
```

```
# DONT TOUCH
t.penup()
t.goto(50, 150)
t.pendown()
t.setheading(0)
# DON'T TOUCH




# SQUARE CODE --- vvvvvvvvvvvvvvvvv

# END SQUARE CODE--- ^^^^^^^^^^^^^^^^




# DON'T TOUCH
t.penup()
t.goto(-200, -50)
t.pendown()
t.setheading(0)
# DON'T TOUCH







# BEGIN STAR CODE --- vvvvvvvvvvvvvvvvv

# END STAR CODE --- ^^^^^^^^^^^^^^^^



# DON'T TOUCH
t.penup()
t.goto(50, -50)
t.pendown()
t.setheading(0)
```

```
# DON'T TOUCH




# INFINITE DESIGN CODE --- vvvvvvvvvvvvvvvvv

# END DESIGN CODE --- ^^^^^^^^^^^^^^^^^
```

```
# DON'T TOUCH
t.penup()
t.goto(-50, 0)
t.pendown()
t.setheading(0)
# DON'T TOUCH
```

```
# ODD-POINTS STAR CODE --- vvvvvvvvvvvvvvvvv
amount_of_sides = 9

# END ODD-POINTS STAR CODE --- ^^^^^^^^^^^^^^^^^


input()
```

Handouts

Do now: think of a question about while loops that hasn't been answered yet.

In-Class Exercises

The class is a lab, so students will use the above starter file and spend the class doing that activity.

The extension assignment is to create the odd-vertices star code. Extension beyond that is the extension of the number-guessing game from lab 1 - adding a high score.

Assignments

The homework is to:

- finish any late homework for 10% off
- finish any remaining work on either of the 2 the labs
- study for the while loops quiz next class

Class 03 - intro to definite loops

Learning Objectives

- Students will be able to write a for loop independently in Python.
- Students will be able to use the loop variable correctly in for loops.
- Students will be able to connect their work with for loops to their understanding of while loops.
- Students will be able to identify the parts of a for loop that distinguish it from a while loop.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.

Materials & Resources

The following source code file:

```
"""
Class 3 Starter Code
For each exercise, pass `end=' '` to your `print` statements, like this:

print("hello", end="")

This makes your output easier to inspect.
"""

# Exercise 1
# Write a for loop that prints each number from 1 to 10.
print("\nExercise 1")
```

```
# Exercise 2
# Write a for loop that prints each number from 1 to 100.
print("\nExercise 2")



# Exercise 3
# Write a for loop that prints each odd number from 1 to 99.
print("\nExercise 3")



# Exercise 4
# Write a for loop that prints the first 10 perfect squares (1, 4, 9, 16,
etc. -- perfect squares have square roots that are whole numbers.)
print("\nExercise 4")



# Exercise 5
# Change only line A so that this program counts from 0 to 99.
print("\nExercise 5")

for i in range(10):
    for j in range(10):
        print(0, end=' ') ### line A


# Exercise 6
# Write a program that asks the user for an integer and then uses a for
loop to print the first 5 multiples of that integer.
print("\nExercise 6")



# Extension Exercises
# 1. Write a program that sums up all of the
#    numbers from 1 to a given number, using
#    a for loop.
#
# 2. Write a program that uses a for loop to
#    determine how many digits are in an integer.
#
# 3. Write a program that uses for loops
#    or while loops to determine if a
```

```
#    number is prime.
# print("Extension Exercises:")
```

Handouts

`while` quiz

1. Write a while loop that counts backwards by ones from 20, starting at 20 and ending at 0.
2. Write a while loop that will never run.
3. Write a while loop that will run forever.

In-Class Exercises

Do Now: think: what does the word "for" mean to you?

The following demos will be done:

```
print("while:")
i = 0
while i < 5:
    print(i)
    i = i + 1

print("for:")
for i in range(5):
    print(i)

for i in range(0, 5, 1):
    print(i)

for i in range(5, 0, -1):
    print(i)

for i in range(0, 100, 2):
    print(i)

for i in range(100, 0, -2):
    print(i)
```

```
for i in range(3):
    print(i)
    for j in range(3):
        print("hip", end=", ")
    print("hooray!")
```

The in-class exercises are contained in the source file that the students receive.

Assignments

Problems 5.3.1, 5.3.2, 5.3.3 from https://runestone.academy/ns/books/published/CS1-Python-Subgoals/for_loop_range.html?mode=browsing

Write your answer to the problem as a private comment on Google Classroom.

Class 04 - definite loops practice i

Learning Objectives

- Students will be able to write for loops independently.
- Students will be able to use their knowledge of while loops to support their learning of for loops.
- Students will be able to independently extend their knowledge of for loops to familiar contexts.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.

Materials & Resources

```
For this lab, you'll be implementing all of the exercises that we did for
while loops, except using for loops.

"""
Definite Loops

Definite loops run a specific amount of times.

Exercise 1:
Write a for loop that prints the numbers 0-4, inclusive.
"""

# # Uncomment these lines by highlighting all of them and pressing Ctrl + /

# print("Exercise 1:")
```

```
"""
Exercise 2:

Write a while loop that prints out the numbers 0-10, inclusive.
"""

# put exercise 2 here.

# print("Exercise 2:")


"""
Exercise 3:

Write a for loop that prints the following output:
a
aa
aaa
aaaa
aaaaa
aaaaaa
"""
# write exercise 3 here.

# print("Exercise 3:")

"""
Exercise 4:

Write a for loop that prints every even number from 0 to 100.
"""

# write exercise 4 here.

# print("Exercise 4:")

"""
Exercise 5:
Write a for loop that prints the first 20 powers of 2.
(1, 2, 4, 8, 16, 32, 64, 128, etc...)
"""

# write exercise 5 here.

# print("Exercise 5:")

"""
```

```
Exercise 6:
Write a short program that takes a number from the user, and counts down
from that number.
"""

# write exercise 6 here.

# print("Exercise 6:")

# number = int(input("What number are we counting down from?"))


"""
Independent practice:

Using the modulo operator (%) and while and if,
write a program that prints out all of the
factors of a number. Here is a useful (!) snippet
of code that prints a factor if it is a factor of
a number.

if number % factor == 0:
  print(factor)

Your program should:
1. Ask the user for a number.
3. Have a for loop that uses factor and number in the conditional
4. Have an if statement in the for loop
5. Have a print statement in the if block
"""

# write your independent practice program below.



"""
Optional challenges:

1. Write a program that asks the user for a number, n,
and prints the first n numbers of the Fibonacci sequence.

2. Write a program that makes a zigzag pattern
in the terminal, like this:
_____
____
___
__
```

```
_

_
__
___
____
_____
____
___
etc.
Hint: You will need 1 while loop and 2 for loops for this.

3. Write another version of your lab #1 with while loops to only give the
user 6 tries to guess the number. Keep track of the user's win/loss
percentage, instead of a high score.
"""

# write your optional challenge programs below.
```

Handouts

In-Class Exercises

Do Now: think of how you could have implemented the first while lab using for loops.

The file contains all of the necessary exercises.

Assignments

5.4.1 Exercises 1-5 from https://runestone.academy/ns/books/published/CS1-Python-Subgoals/loops-flr-p1.html?mode=browsing

Write each answer as a private comment on Google Classroom.

Class 04 - definite loops practice 2

Learning Objectives

- Students will be able to apply their knowledge of for loops to Python with Turtle scenarios.
- Students will be able to independently troubleshoot and debug their Python with Turtle code.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

Students will be coding using this file as starter code:

```
"""
Definite Loops (for) Lab II

Use this starter code to create the design goal.

Don't change the setup code OR the code sandwiched between comments that
say "DON'T TOUCH!"

"""

# setup - do not change!
import turtle

t = turtle.Turtle()
```

```
s = turtle.Screen()
s.setup(1.0, 1.0)
t.penup()
t.speed(0)

t.goto(-200, 150)
t.pendown()
t.setheading(0)



# BEGIN TRIANGLE CODE --- vvvvvvvvvvvvvvvvv


# END TRIANGLE CODE --- ^^^^^^^^^^^^^^^^




# DONT TOUCH
t.penup()
t.goto(50, 150)
t.pendown()
t.setheading(0)
# DON'T TOUCH




# SQUARE CODE --- vvvvvvvvvvvvvvvvv

# END SQUARE CODE--- ^^^^^^^^^^^^^^^^






# DON'T TOUCH
t.penup()
t.goto(-200, -50)
t.pendown()
t.setheading(0)
# DON'T TOUCH
```

```
# BEGIN STAR CODE --- vvvvvvvvvvvvvvvvv

# END STAR CODE --- ^^^^^^^^^^^^^^^^^



# DON'T TOUCH
t.penup()
t.goto(50, -50)
t.pendown()
t.setheading(0)
# DON'T TOUCH




# INFINITE DESIGN CODE --- vvvvvvvvvvvvvvvvv

# END DESIGN CODE --- ^^^^^^^^^^^^^^^^^



# DON'T TOUCH
t.penup()
t.goto(-50, 0)
t.pendown()
t.setheading(0)
# DON'T TOUCH
```

```
# ODD-POINTS STAR CODE --- vvvvvvvvvvvvvvvvv
amount_of_sides = 9

# END ODD-POINTS STAR CODE --- ^^^^^^^^^^^^^^^^^



input()
```

Handouts

In-Class Exercises

Do Now: how do for loops make geometry in Turtle easier?

The lab file contains all of the exercises.

Assignments

The homework is to:

- finish any late homework for 10% off
- finish any remaining work on either of the 2 the labs
- study for the for loops quiz next class

Class 06 - intro to lists

Learning Objective[s]

- Students will be able to use lists to store collections of data in Python.
- Students will be able to use the documentation/reference sheet for lists to write their own practice exercises for lists.
- Students will be able to differentiate between the list methods that return a value and the list methods that only mutate a list.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

These are the questions that will be posted on Google Classroom for students to work on independently, in their own file titled `lists.py` .

0. (kind of long) - Using the resources here, demonstrate your understanding of each list method by implementing each method once with a sample list.
1. Write a line of python that creates a variable a and sets it equal to a list holding the first 5 numbers of the school's phone number.
2. Write a line of Python that creates a variable b and sets it equal to a list with one integer, one string, and one float.

3. In one sentence, explain why program A prints “Boop!”
4. Modify line 1 of Program A so that it prints “Beep!”

Program A

```
my_list = [1, 0, 1, 0, 4, 5]
if my_list[0] < my_list[2]:
  print(“Beep!”)
else:
  print(“Boop!”)
```

5. What is the output of line A in program B?
6. What is the output of line B in program B?
7. What is the output of line C in program B?

Program B

```
my_list = [1, 3, 4]
my_list.append(0)
my_list.append(3)
print(my_list)    # **** line A ****
my_list = my_list + my_list
print(my_list)    # **** line B ****
print(my_list[0]) # **** line C ****
```

8. For each line of program C, write what would happen if you entered it into Python’s REPL.

Program C

```
a = [1, 2, 3]
print(a[0])
print(a[-1])
a[0] > a[2]
a[4]
a.append(a[0])
a.append(a[0])
a
len(a)
a.pop(0)
a
```

```
a.remove(3)
a
```

9. Extension: write a program that creates a grocery list for a user. The program should use a while loop to add items to the grocery list until the user types `done`, and then the list should be printed for the user.

Handouts

`for` quiz

1. Write a for loop that starts at 1970 and goes until 2020, inclusive of both years.
2. Write a for loop that will never run.
3. Write a for loop that checks all of the possible factors of a number. Remember that `3 % 2 == 1` and `4 % 2 == 0`.

In-Class Exercises

Do Now: There are lists in python. What do you think they do?

The following demo code will be run:

```
names = []
names.append("Luis")
names.append("Diego")

print(names)

print(names[0])

print(names[1])

topname = names.pop()
print(topname)

topname = names.pop()
print(topname)

for i in range(10):
```

```
  names.append(str(i)*5)

print(names[0])


print(names[-1])


print(names[2:4])


print(names[0:-1])


print(names.index("11111"))


names = names + names

print(names)

for n in names:
  print(n)



while len(names) > 0:
  print(names.pop())
```

The file above contains all of the activities for the in-class exercises.

Assignments

Homework from Runestone, with answers attached to Google Classroom as a screenshot:

https://runestone.academy/ns/books/published/CS1-Python-Subgoals/loops-cc-p1.html?mode=browsing

Class 07 - lists + indefinite loops practice

Learning Objective[s]

- Students will be able to independently write programs that utilize lists and while loops.
- Students will be able to write several lines of code inside of a while loop and keep track of the operations happening.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.

Materials & Resources

```
# Lists + While lab

# Task 1: write a program that asks the user for integers to be put into a
list. Convert every number that the user enters into an integer and then
append the integers to a list. When the user types "done", stop adding to
the list.

# Task 2: write a while loop that goes through the list that the user
generated and prints out the maximum value.

# Task 3: write a while loop that goes through the list that the user
generated and prints out the minimum value.

# Extension task: See if you can use your knowledge of loops and lists to
sort the list of integers from smallest to greatest.
```

Handouts

In-Class Exercises

Do Now: What program in Python do you think would use a list and a while loop?

The in-class exercise is the file described in the Materials section of the lesson.

Assignments

Exercises from Runestone, with answers posted as a private comment on Google Classroom.

https://runestone.academy/ns/books/published/CS1-Python-Subgoals/while_loops_sentinel.html?mode=browsing

Class 08 - lists + definite loops practice

Learning Objective[s]

- Students will be able to write programs that use for loops and lists together.
- Students will be able to write programs with significant code inside of a for loop.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

```
# Lists + For lab
"""
Ice Cream Combinations
The person using your program runs a frozen yogurt shop. They would like to
make a banner sharing how many combinations of frozen yogurt, mix-ins, and
toppings are available. They hire you to write a program to find out all of
the available combinations. Combinations look like this:

Vanilla
Chocolate with blueberries and chocolate sprinkles
Strawberry with boba
"""


# Task 1: Create 4 empty lists:
# flavors, mixins, toppings, and all_combinations
```

```
# Task 2: using .append(), populate your first 3 lists.
# Flavors: vanilla, chocolate, strawberry, butter pecan, and mango
# Mix-ins: blueberries, boba, strawberries, granola, oreo
# Toppings: rainbow sprinkles, chocolate sprinkles, caramel syrup,
chocolate syrup


# Task 3: Use a for loop to add each flavor to the all_combinations list.



# Task 4: Use two different for loops to add all combinations of 1 flavor
and 1 mixin to the all_combinations list.


# Task 5: Use three different for loops to add all combinations of 1 of
each ingredient to the all_combinations list.


# Task 6: Go back and make sure that the combinations in your list look
good when printed.

# Task 7: print all combinations with a comma in between them. Print out
how many combinations there are.

# Extension task 1: use list.index(element) and the following pricing lists
to add a price to each combination:
flavor_price = [2, 1.8, 2.2, 3, 1.5]
mixin_price = [2, 1, 2, 2, 1]
topping_price = [1, 1, 2, 2]

# Extension task 2: Expand your program to work with user input - use while
loops to accomplish the user input. Do not worry about prices with this
extension.
```

Handouts

In-Class Exercises

Do Now: what type of program do you think would use a for loop with a list?

Lab included above.

Assignments

Exercises from Runestone, with answers posted as private comments in Google Classroom

https://runestone.academy/ns/books/published/CS1-Python-Subgoals/for_loops_lists.html?mode=browsing

Class 09 - lists + loops practice

Learning Objective[s]

- Students will be able to write a program without programming scaffolding that uses while loops, for loops, and lists.
- Students will be able to make design choices for a program based on design constraints.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

```
"""
Blankman

You know the game where people guess a word, letter by letter? Implement
your own version of it. It can work however you want it to - however, these
are the requirements:

1. The user should guess letters by typing them and pressing enter
2. The program should  display the word, with blanks, each time the user is
about to guess a new letter.

To start with, the word that is being guessed is "programming".
"""
```

```
# Extension: Make your program resistant to many different kinds of input
(entire words, integers for guesses, etc.).
```

Handouts

`lists` quiz

1. In the program below, `mylist` can change, but the rest of the program cannot. Write a program that iterates through the list to find the minimum and maximum of the list. You can use a for loop or a while loop.

```
mylist = [3, 8, 5, 3, 5, 78, 3, 7, 4, 2, 4, 6]

## type your program here
```

In-Class Exercises

Do Now: how would you implement blankman (hangman) in Python?

Teacher demo with working code to give inspiration for how program will work.

File found above.

Assignments

Finish blankman.

Class 10 - lists + loops lab time + review

Learning Objective[s]

- There are no explicit learning objectives, other than the sum of all of the learning objectives already stated for this unit.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

Practice Assessment

1. Write a program that asks the user, "what is the password?" and only prints "correct!" if they type "testing123".
2. Write two programs for each of the below bullet points - one program should use a for loop, the other should use a while loop.

- Given a list `a = [1, 2, 3, 4, 5]`, write a program that prints the first five multiples of each number in `a`.
- Given a list `b` that contains all of the integers between 0 and 99, write a program that goes through `b` and prints out whether each number in `b` is prime.

3. Write out all of the list methods that you remember. What do they each do?

4. What does this program do? Be very descriptive.

```
values = []
alphabet = list("abcdefghijklmnopqrstuvwxyz")

for j in range(26):
  values.append(0)

sample = "the quick brown fox jumps over the lazy dog"

for letter in sample:
  values[alphabet.index(letter)] += 1


for i in range(len(values)):
  print(alphabet[i], values[i])
```

Handouts

In-Class Exercises

Assignments

Class 11 - lists + loops summative assessment

Learning Objective[s]

- There are no explicit learning objectives, other than the sum of all of the learning objectives already stated for this unit.

Standards

- **9-12.CT.4** - Implement a program using a combination of student-defined and third-party functions to organize the computation.
- **9-12.CT.8** - Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.
- **9-12.CT.9** - Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior.
- **9-12.CT.10** - Collaboratively design and develop a program or computational artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Materials & Resources

Summative Assessment

1. Give an example of a task that a **while** loop could accomplish, but not a **for** loop. Your answer should **not** be in code. Feel free to use an example from one of the labs we did.

2. Give an example of a task that would be simpler to implement using a **for** loop instead of a **while** loop. Your answer should **not** be in code. Feel free to use an example from one of the labs we did.

3. Write a program that uses a while loop to calculate the **sum of the first ten even numbers**.

4. Write a program that uses a **for loop** to do the **same task as #3** - calculate the sum of the first ten even numbers.

Program A

```
colors = ["red", "orange", "yellow", "green", "blue", "purple"]
vehicles = ["car", "bike", "skateboard", "rollerskates"]
```

5. Write a program that prints out every combination of color and vehicle from Program A. Your program can print it out in any order - one possible order would be:

```
red car
red bike
red skateboard
red rollerskates
orange car
orange bike
orange skateboard
orange rollerskates
yellow car
```

...and so on.

Questions 6, 7, and 8 have to do with Program B. Read it closely.

Program B

```
user_name = input("what is your name?")

alphabet = "abcdefghijklmnopqrstuvwxyz"
```

```
finished = []

for l in user_name:       # flag A
  a = 0
  while alphabet[a] != l: # flag B
    a += 1
  finished.append(a)

# flag

print("Output:")
for b in finished:
  print(b, end=" ")
```

6. What does the for loop marked with flag A do?

7. What does the while loop marked with flag B do?

8. What does this entire program do?

9. If `finished = [10, 9, 3, 3]` were inserted at the line marked `#flag C`, what would this program output?

In-Class Exercises

Assignments