1. Assign the value 7 to the variable guess\_me. Then, write the conditional tests (if, else, and elif) to print the string 'too low' if guess\_me is less than 7, 'too high' if greater than 7, and 'just right' if equal to 7.

4.1 Assign the value 7 to the variable guess\_me. Then, write the conditional tests (if,

# else, and elif) to print the string 'too low' if guess\_me is less than 7, 'too high' if

# greater than 7, and 'just right' if equal to 7.

section = '4.1'

print\_header(section)

guess\_me = 7

if guess\_me < 7:

print("too low")

elif guess\_me > 7:

print("too high")

else:

print("just right")

print\_footer(section)

2. Assign the value 7 to the variable guess\_me and the value 1 to the variable start. Write a while loop that compares start with guess\_me. Print too low if start is less than guess me. If start equals guess\_me, print 'found it!' and exit the loop. If start is greater than guess\_me, print 'oops' and exit the loop. Increment start at the end of the loop.

section = '4.2'

print\_header(section)

guess\_me = 7

start = 1

while True:

if start < guess\_me:

print('too low')

elif start == guess\_me:

print('found it!')

break

else:

print('oops')

break

start += 1

print\_footer(section)

3. Print the following values of the list [3, 2, 1, 0] using a for loop.

section = '4.3'

print\_header(section)

numbers = [3, 2, 1, 0]

for num in numbers:

print(num)

print\_footer(section)

4. Use a list comprehension to make a list of the even numbers in range(10)

section = '4.4'

print\_header(section)

even\_numbers = [num for num in range(10) if num % 2 == 0]

print(even\_numbers)

print\_footer(section)

5. Use a dictionary comprehension to create the dictionary squares. Use range(10) to return the keys, and use the square of each key as its value.

section = '4.5'

print\_header(section)

limit = 10

squares = {num: num \* num for num in range(limit)}

print(squares)

print\_footer(section)

6. Construct the set odd from the odd numbers in the range using a set comprehension (10).

7. Use a generator comprehension to return the string 'Got ' and a number for the numbers in range(10). Iterate through this by using a for loop.

section = '4.7'

print\_header(section)

limit = 10

string\_generator = ('Got ' + str(num) for num in range(limit))

for item in string\_generator:

print(item)

print\_footer(section)

8. Define a function called good that returns the list ['Harry', 'Ron', 'Hermione'].

9. Define a generator function called get\_odds that returns the odd numbers from range(10). Use a for loop to find and print the third value returned.

section = '4.9'

print\_header(section)

limit = 10

get\_odds = (num for num in range(limit) if not num % 2 == 0)

count = 0

for num in get\_odds:

if count == 2:

print(num)

break

count += 1

print\_footer(section)

10. Define an exception called OopsException. Raise this exception to see what happens. Then write the code to catch this exception and print 'Caught an oops'.

print\_header(section)

class OopsException(Exception):

pass

def with\_exception(a):

if a < 0:

raise OopsException(a)

try:

with\_exception(-1)

except OopsException as err:

print('Caught an oops')

print\_footer(section)

11. Use zip() to make a dictionary called movies that pairs these lists: titles = ['Creature of Habit', 'Crewel Fate'] and plots = ['A nun turns into a monster', 'A haunted yarn shop'].

section = '4.12'

print\_header(section)

titles = ['Creature of Habit', 'Crewel Fate']

plots = ['A nun turns into a monster', 'A haunted yarn shop']

movies = {}

for title, plot in zip(titles, plots):

movies[title] = plot

# or movies = dict(zip(titles, plots))

print(movies)

print\_footer(section)