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.

Sol: python:

guess\_me = 7

if guess\_me < 7:

print('too low')

elif guess\_me > 7:

print('too high')

else:

print('just right')

just right

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.

Sol: python:

guess\_me = 7

start = 1

while True:

if start < guess\_me:

print('too low')

elif start == guess\_me:

print('found it!')

break

elif start > guess\_me:

print('oops')

break

start += 1

output:

too low

too low

too low

too low

too low

too low

found it!

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

Sol: python:

my\_list = [3, 2, 1, 0]

for value in my\_list:

print(value)

3

2

1

0

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

Sol: python:

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

print(even\_numbers)

[0, 2, 4, 6, 8]

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.

Sol: python:

squares = {num: num\*\*2 for num in range(10)}

print(squares)

{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}

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

Sol: python:

odd = {num for num in range(10) if num % 2 != 0}

print(odd)

{1, 3, 5, 7, 9}

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.

Sol: python:

generator = ('Got ' + str(num) for num in range(10))

for item in generator:

print(item)

Got 0

Got 1

Got 2

Got 3

Got 4

Got 5

Got 6

Got 7

Got 8

Got 9

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

Sol: python:

def good():

return ['Harry', 'Ron', 'Hermione']

# Call the function

result = good()

print(result)

['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.

Sol: python:

def get\_odds():

for num in range(10):

if num % 2 != 0:

yield num

count = 0

for value in get\_odds():

count += 1

if count == 3:

print(value)

break

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'.

Sol: python:

class OopsException(Exception):

pass

raise OopsException

try:

raise OopsException

except OopsException:

print('Caught an oops')

Traceback (most recent call last):

File "<stdin>", line 7, in <module>

\_\_main\_\_.OopsException

Caught an oops

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'].

Sol: python:

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

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

movies = dict(zip(titles, plots))

print(movies)

{'Creature of Habit': 'A nun turns into a monster', 'Crewel Fate': 'A haunted yarn shop'}