1. Assign the value 7 to the variable guess\_me. Then, write the conditional tests (if, else, and elif) to

print the string &#39;too low&#39; if guess\_me is less than 7, &#39;too high&#39; if greater than 7, and &#39;just right&#39; if equal to 7.

**def** guess\_me(guess\_me):

**if** guess\_me **<** 7:

print('too Low')

**elif** guess\_me **>** 7:

print('too High')

**else**:

print('just Right')

guess\_me(guess\_me**=**7)

guess\_me(guess\_me**=**5)

guess\_me(guess\_me**=**15)

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 &#39;found it!&#39; and exit the loop. If start is greater than guess\_me, print &#39;oops&#39; and exit

the loop. Increment start at the end of the loop.

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

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

in\_list **=** [3,2,1,0]

**for** ele **in** in\_list:

print(ele)

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

print([x **for** x **in** range(10**+**1) **if** x**%2**==0 ])

print(dict([(x,pow(x,2)) **for** x **in** range(10)]))

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.

print({x **for** x **in** range(10) **if** x**%2** !=0})

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

gen\_com **=** ('Got\_'**+**str(x) **for** x **in** range(10))

**for** ele **in** gen\_com:

print(ele, end**=**' ')

7. Use a generator comprehension to return the string &#39;Got &#39; and a number for the numbers in range(10). Iterate through this by using a for loop.

gen\_com **=** ('Got\_'**+**str(x) **for** x **in** range(10))

**for** ele **in** gen\_com:

print(ele, end**=**' ')

8. Define a function called good that returns the list [&#39;Harry&#39;, &#39;Ron&#39;, &#39;Hermione&#39;].

**def** good():

x **=** ['Harry', 'Ron', 'Hermione']

**return** x

print(good())

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.

**def** get\_odds():

output **=** []

**for** ele **in** range(10):

**if** ele**%2** != 0:

output**.**append(ele)

**yield** output

next(get\_odds())[2]

10. Define an exception called OopsException. Raise this exception to see what happens. Then write

the code to catch this exception and print &#39;Caught an oops&#39;.

**class** OopsException(Exception):

**pass**

**def** test(input):

**if** input **<**0:

**raise** OopsException(a)

**try**:

test(**-**100)

**except** Exception **as** e:

print('Caught in Oops ->',e)

11. Use zip() to make a dictionary called movies that pairs these lists: titles = [&#39;Creature of Habit&#39;,

&#39;Crewel Fate&#39;] and plots = [&#39;A nun turns into a monster&#39;, &#39;A haunted yarn shop&#39;].

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

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

output **=** dict(zip(titles,plots))

print(output)