#### 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.

I/P:

**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)

O/P:

just Right

too Low

too High

#### 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

I/P:

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

O/P:

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.

I/P:

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

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

print(ele)

O/P:

3

2

1

0

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

I/P:

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

O/P:

[0, 2, 4, 6, 8, 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.

I/P:

*# Method 1*

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

*# Method 2*

print({x:x**\*\***2 **for** x **in** range(10)})

O/P:

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

{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).

I/P:

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

O/P:

{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

I/P:

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

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

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

O/P:

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']

I/P:

**def** good():

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

**return** x

print(good())

O/P:

['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.

I/P:

**def** get\_odds():

output **=** []

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

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

output**.**append(ele)

**yield** output

next(get\_odds())[2]

O/P:

5

#### 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'.

I/P:

**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)

O/P:

Caught in Oops -> name 'a' is not defined

#### 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'].

I/P:

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

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

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

print(output)

O/P:

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