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.

**Ans.**

**guess\_me = 7**

**if guess\_me < 7:**

**print('too low')**

**elif guess\_me > 7:**

**print('too high')**

**else :**

**print('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 equalsguess\_me, ‘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.

**Ans.**

**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**

**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.

**Ans.**

**numbers = [3, 2, 1, 0]**

**for i in numbers:**

**print(i)**

**Output:**

3

2

1

0

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

**Ans.**

even\_numbers = [num for num in range(1,10) if num % 2 == 0]

print(even\_numbers)

**Output:**

[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.

**Ans.** squares = {x: x\*\*2 for x in range(10)}

print(squares)

**Output:**

{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).

**Ans.** odd = {n for n in range(1, 11) if n % 2 == 1}

odd

**Output:**

{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.

**Ans.**

generator = ('Got {}'.format(num) for num in range(10))

for item in generator:

print(item)

**Output:**

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’].

**Ans.**

def good():

  return ['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.

**Ans.**

def get\_odds():

for i in range(1, 10, 2):

yield i

for i, odd in enumerate(get\_odds()):

if i == 2:

print(odd)

break

**Output:**

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’.

**Ans.**

# Define the OopsException

class OopsException(Exception):

pass

# Raise the OopsException to see what happens

try:

raise OopsException

except OopsException:

# Catch the OopsException and print a message

print('Caught an oops')

**Output:**

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’].

**Ans.**

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

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

movies = dict(zip(titles, plots))

movies

**Output:**

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