**ADITYA AMIN ASSIGN : 17**

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.

guess\_me = 7

# Write the conditional tests

if guess\_me < 7:

print('too low')

elif guess\_me > 7:

print('too high')

else:

print('just right')

1. 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.

# Assign the value 7 to the variable guess\_me

guess\_me = 7

# Assign the value 1 to the variable start

start = 1

# Write the while loop

while start <= guess\_me:

if start < guess\_me:

print('too low')

elif start == guess\_me:

print('found it!')

break

else:

print('oops')

break

start += 1

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

A = [3,2,1,0]

for I in A:

Print(i)

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

# Use list comprehension to create a list of even numbers in range(10)

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

# Print the list of even numbers

print(even\_numbers)

1. 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.

# Use dictionary comprehension to create a dictionary of squares

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

# Print the dictionary of squares

print(squares)

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

# Use set comprehension to create a set of odd numbers in range(10)

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

# Print the set of odd numbers

print(odd)

1. 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.

# Use generator comprehension to generate 'Got ' + number for numbers in range(10)

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

# Iterate through the generator using a for loop

for item in gen:

print(item)

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

def good():

"""This function returns a list of three names."""

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

# Call the good function and store the returned list in a variable

my\_list = good()

# Print the returned list

print(my\_list)

1. 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():

"""This generator function yields odd numbers from range(10)."""

for num in range(10):

if num % 2 != 0:

yield num

# Call the generator function and iterate through the values

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

if i == 2:

print("The third odd number is:", num)

break

1. 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'.

# Define the custom exception

class OopsException(Exception):

pass

try:

# Raise the custom exception

raise OopsException("Something went wrong!")

except OopsException as e:

# Catch the custom exception and print the message

print("Caught an oops:", e)

1. 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'].

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

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

# Use zip() to create a dictionary

movies = dict(zip(titles, plots))

# Print the resulting dictionary

print(movies)