EXERCISES 6 , 7, 9

Date : 23.11.20

6)

AIM:

To write and run a Python program to fill in the desired output.

PROGRAM:

# Create a tuple, also called tuple packing.

numbers = 1, 2

print(numbers)

(1, 2)

# Create tuple with paranthesis.

numbers = (1, 2, 3)

print(numbers)

(1, 2, 3)

# Create an empty tuple.

numbers = ()

print(numbers)

()

# Create a tuple with one item. Note that the trailing comma is necessarynumbers = 1,

print(numbers)

1

# Create a tuple with heterogenous items.

random\_tuple = "Hey", (1, 2), 1, ["you"]

print(random\_tuple)

('Hey', (1, 2), 1, ['you'])

# Create tuple with tuple() constructor.

numbers = tuple()

print(numbers)

()

numbers = tuple([1, 2]) # Takes any sequence as input

print(numbers)

(1,2)

#### Methods on tuples #####

# Get length of list by using len() method.

numbers = 5, 8, 8

print(len(numbers))3

# Get index of an element using the index() method.

numbers = 5, 8, 8

print(numbers.index(8))

1

# Count occurences of an item in a tuple.

numbers = 5, 8, 8

print(numbers.count(8))

eggs = ('hello', 42, 0.5)

eggs[0]

'hello'

hello

eggs[1:3]

(42, 0.5)

len(eggs)

3# Access elements of a tuple by indexing.

str\_tuple = "hey", "there!", "how", "are", "you?"

print(str\_tuple[0])

hey

print(str\_tuple[len(str\_tuple) - 1])

you?

print(str\_tuple[-1])

you?

# Slicing a tuple.

str\_tuple = "hey", "there!", "how", "are", "you?"

print(str\_tuple[2:])

('how', 'are', 'you?')

print(str\_tuple[:2])

('hey', 'there!')

print(str\_tuple[-3:])

('how', 'are', 'you?')print(str\_tuple[:-3])

('hey', 'there!')

print(str\_tuple[1:4])

('there!', 'how', 'are')

# Get a copy of the tuple by slicing.

print(str\_tuple[:])

('hey', 'there!', 'how', 'are', 'you?')

# Concatenate tuples.

numbers = (1, 2)

strings = ("Hey", "there")

print(numbers + strings)

(5, 8, 8, 'Hey', 'there')

(1, 2, "Hey", "there")

# Looping through tuple using 'in'.

numbers = 1, 2

for number in numbers:

print(number)

1,2

12# Check if element is present in tuple.

numbers = 1, 2

print(1 in numbers)

True

print(5 in numbers)

False

# Tuple packing.

# We are packing two items 1 and 2 into the tuple.

numbers = 1, 2

# Tuple sequence unpacking.

# Number of variables used has to be same as the number of items in the

tuple.

# Unpacking the tuple and assigning its items to x and y.

x, y = numbers

# Note that this is also packing the args as a tuple which gets unpacked as

the print method's arguments.

print(x, y)

12

RESULT:

Thus, we run a Python program to fill in the desired output Successfully.EXERCISES - 7

7)

AIM:

To write and run a Python program to fill in the desired output.

PROGRAM:

primes = [2, 3, 5, 7, 11]

print(primes)

# Output: [2, 3, 5, 7, 11]

tems = ['cake', 'cookie', 'bread']

total\_items = items + ['biscuit', 'tart']

print(total\_items)

# Output:['cake', 'cookie', 'bread', 'biscuit', 'tart']

orders = ['daisies', 'periwinkle']

orders.append('tulips')

print(orders)

# Result: ['daisies', 'periwinkle', 'tulips']

owners\_names = ['Jenny', 'Sam', 'Alexis']

dogs\_names = ['Elphonse', 'Dr. Doggy DDS', 'Carter']

owners\_dogs = zip(owners\_names, dogs\_names)

print(list(owners\_dogs))

# Result: [('Jenny', 'Elphonse'), ('Sam', 'Dr.Doggy DDS'), ('Alexis', 'Carter')

items = [1, 2, 3, 4, 5, 6]

print(items[:4]) #Output: [1, 2, 3, 4]

print(items[2:]) #Output: [3, 4, 5, 6]knapsack = [2, 4, 3, 7, 10]

size = len(knapsack)

print(size) # Output: 5

cnt = knapsack.count(7)

print(cnt) # Output: 1

exampleList = [4, 2, 1, 3]

exampleList.sort()

print(exampleList)

# Output: [1, 2, 3, 4]

soups = ['minestrone', 'lentil', 'pho', 'laksa']

soups[-1] # output: 'laksa'

soups[-3:] # output: 'lentil', 'pho', 'laksa'

soups[:-2] # output: 'minestrone', 'lentil'

RESULT:

Thus, we run a Python program to fill in the desired output Successfully.EXERCISES - 9

9)

AIM:

To write and run a Python program to fill in the desired output.

PROGRAM:

print('\n—dictionaries')

#Output: -- dictionaries

d = {'a': 1, 'b': 2}

print(d['a'])

#Output: 1

del d['a']

# iterate

d = {'a': 1, 'b': 2}

for key, value in d.items():

print(key, ':', value)

for key in d:

print(key, d[key])

# d.fromkeys(iterable[,value=None]) -> dict: with keys from iterable and all

same value

d = d.fromkeys(['a', 'b'], 1)

print(d)

#Output: {'a': 1, 'b': 1}

# d.clear() -> removes all items from d

d = {'a': 1, 'b': 2}

d.clear()

print(d)

#Output: {}# d.items() -> list: copy of d's list of (key, item) pairs

d = {'a': 1, 'b': 2}

print(d.items())

#Output: [('a', 1), ('b', 2)]

# d.keys() -> list: copy of d's list of keys

d = {'a': 1, 'b': 2}

print(d.keys()) #Output: ['a', 'b']

# d.values() -> list: copy of d's list of values

d = {'a': 1, 'b': 2}

print(d.values()) #Output: [1, 2]

RESULT:

Thus, we run a Python program to fill in the desired output Successfully.