# **Splitting Strings**

**Model Answers**

**Challenge 1**

**Write a program that takes a single user input of 3 integers separated by commas and then outputs the average of the numbers.**

# MorePracticeWithStringsChallenge1.py

# Challenge 1

# Author: A. N. Other

# date: November 2016

user\_numbers = input("Please enter 3 integers separated by commas.\n\n")

split\_input = user\_numbers.split(",")

#turn the list of strings into a list of int

int\_list = [int(i) for i in split\_input]

#calculate average of items in a list

calc\_average = sum(int\_list)/len(int\_list)

print(calc\_average)

'''

#asertions

Input: 3,4,5

Output: 4.0

'''

**Challenge 2**

**Write a program that takes a single user input of any number of integers separated by commas and then outputs the sum of the numbers.**

# MorePracticeWithStringsChallenge2.py

# Challenge 2

# Author: A. N. Other

# date: November 2016

user\_numbers = input("Please enter any number of integers separated by commas.\n\n")

split\_input = user\_numbers.split(",")

#turn the list of strings into a list of int

int\_list = [int(i) for i in split\_input]

#calculate sum of items in a list

calc\_sum = sum(int\_list)

print(calc\_sum)

'''

#assertions

Input: 4,6,7,8,2,3,1

Output: 31

'''

**Challenge 3**

**Write a program that randomly jumbles the words in a stored sentence then displays the output. Research and use the Python shuffle function.**

# MorePracticeWithStringsChallenge3.py

# Challenge 3

# Author: A. N. Other

# date: November 2016

from random import shuffle

sentence = "this is a sentence that needs to be shuffled."

#split and shuffle sentence

split\_sentence = sentence.split()

shuffle(split\_sentence)

#print sentence

print(split\_sentence)

'''

#assertions

Input: sentence = "this is a sentence that needs to be shuffled."

Output: ['a', 'shuffled.', 'sentence', 'to', 'this', 'is', 'be', 'needs', 'that']

'''

**Challenge 4**

**Write a program to get a string made of the first 2 and the last 2 words from a given a sentence. If the sentence contains less than 2 words, return instead the empty string.**

# MorePracticeWithStringsChallenge4.py

# Challenge 4

# Author: A. N. Other

# date: November 2016

sentence = "This is a sentence from which to make another sentence."

def get\_new\_sentence(sentence):

split\_sentence = sentence.split()

if len(split\_sentence) < 2:

return []

else:

list\_2 = []

list\_2.append(split\_sentence[0])

list\_2.append(split\_sentence[1])

list\_2.append(split\_sentence[-2])

list\_2.append(split\_sentence[-1])

return list\_2

print(get\_new\_sentence(sentence))

'''

#assertion

Input: sentence = "This is a sentence from which to make another sentence."

Output: ['This', 'is', 'another', 'sentence.']

'''

**Challenge 5**

**Write a program that takes a list of words and returns the length of the longest one.**

# MorePracticeWithStringsChallenge5.py

# Challenge 5

# Author: A. N. Other

# date: November 2016

sentence = "This is a sentence containing words of various length."

sentence\_split = sentence.split()

def get\_longest\_word(sentence\_split):

longest\_word = ""

for count in range (0, len(sentence\_split)):

if len(sentence\_split[count]) > len(longest\_word):

longest\_word = sentence\_split[count]

return longest\_word

print(get\_longest\_word(sentence\_split))

'''

#assertion

Input: sentence = "This is a sentence containing words of various length."

Output: containing

'''

**Challenge 6**

**Write a program to remove the nth word from a string containing words separated by spaces.**

# MorePracticeWithStringsChallenge6.py

# Challenge 6

# Author: A. N. Other

# date: November 2016

sentence = "This is a sentence containing words of various length."

sentence\_split = sentence.split()

print("Please enter a number to remove between 1 and ", len(sentence\_split), "\n\n")

index = int(input())

sentence\_split.pop(index - 1)

print(sentence\_split)

'''

#assertion

Input: sentence = "This is a sentence containing words of various length."

4

Output: ['This', 'is', 'a', 'containing', 'words', 'of', 'various', 'length.']

'''

**Challenge 7**

**Write a program that accepts a comma separated sequence of words as input and prints the unique words.**

# MorePracticeWithStringsChallenge7.py

# Challenge 7

# Author: A. N. Other

# date: November 2016

list\_of\_words = "red,orange,yellow,green,blue,indigo,violet,red,green"

list\_split = list\_of\_words.split(",")

unique\_list = set(list\_split)

print(unique\_list)

'''

#assertions

Input: list\_of\_words = "red,orange,yellow,green,blue,indigo,violet,red,green"

Output: {'green', 'indigo', 'violet', 'red', 'orange', 'yellow', 'blue'}

'''

**Challenge 8**

**Write a program that reverses a string if it's length is a multiple of 4.**

# MorePracticeWithStringsChallenge8.py

# Challenge 8

# Author: A. N. Other

# date: November 2016

sentence = "This is a string."

def get\_reversed\_string(sentence):

if len(sentence) % 4 == 0:

sentence\_reversed = "".join(reversed(sentence))

return sentence\_reversed

else:

return sentence

print(get\_reversed\_string(sentence))

'''

#assertions

Input: sentence = "This is a string."

Output: This is a string.

Input: sentence = "This is a string"

Output: gnirts a si sihT

'''

**Challenge 9**

**Write a program that implements an all lowercase Caesar cipher. If you wish to use a list use the list shown here:**

**eng\_dict = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']**

**Here is a sample output from the application:**

* **=> Please enter the text that you wish to encrypt or decrypt, followed by a comma and the integer key.**
* **=> hellothereauckland,3**
* **=> khoorwkhuhdxfndqg**

**Assume that messages use lowercase characters only with no punctuation.**

# MorePracticeWithStringsChallenge9.py

# Challenge 9

# Author: A. N. Other

# date: November 2016

eng\_dict = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']

user\_input = input("Please enter the text that you wish to encrypt or decrypt, followed by a comma and the integer key.\n\n")

split\_user\_input = user\_input.split(",")

sentence = split\_user\_input[0]

offset = int(split\_user\_input[1])

def get\_cipher(sentence, offset):

result = ""

for count in range(0, len(sentence)):

num = ord(sentence[count])

num += offset

if num > ord('z'):

num -= 26

elif num < ord('a'):

num += 26

letter = chr(num)

result += letter

return result

print(get\_cipher(sentence, offset))

'''

#assertion

Input: helloworld,6

Output: nkrrucuxrj

'''

**Challenge 10**

**A stock broker needs an application that allows her to buy or sell shares using a single input statement. Here is a sample program output.**

* **=> sell Bunzle,10**
* **=> done! Bunzle shares remaining: 22**
* **=> buy Prudential,47**
* **=> done! Prudential shares remaining: 107**

**Assume that the application stores the following companies and their shares:**

1. **Bunzle: 32**
2. **Prudential: 22**
3. **Unilever: 60**

# MorePracticeWithStringsChallenge10.py

# Challenge 10

# Author: A. N. Other

# date: November 2016

bunzle = 32

prudential = 22

unilever = 60

def get\_trade(company, transaction, share\_num):

if transaction == "sell":

company -= share\_num

return company

else:

company += share\_num

return company

transaction = input("Please enter trade type then a space followed by the share name a commas and the number.\n\n")

input\_split = transaction.split(" ")

second\_split = input\_split[1].split(",")

transaction = input\_split[0]

co = second\_split[0]

share\_num = int(second\_split[1])

if co.lower() == "bunzle":

bunzle = get\_trade(bunzle, transaction, share\_num)

print("done! Bunzle shares remaining: ", bunzle)

elif co.lower() == "prudential":

prudential = get\_trade(prudential, transaction, share\_num)

print("done! Prudential shares remaining: ", prudential)

else:

unilever = get\_trade(unilever, transaction, share\_num)

print("done! Unilever shares remaining: ", unilever)

'''

#assertion

Input: 32 sell 10

Output: done! Bunzle shares remaining: 22

Input: buy prudential,47

Output: done! Prudential shares remaining: 69

'''

**Challenge 11**

**The requirements for this final challenge are much less detailed than before. This allows you to design the format of the user input.**

**Write a program that takes single user input to calculate any from: area/perimeter/volume of different shapes/solid objects, given the appropriate parameters. You can choose which of the requirements to implement.**

**It would be a sensible approach to reuse existing code that you wrote earlier in the course.**

# MorePracticeWithStringsChallenge11.py

# Challenge 11

# Author: A. N. Other

# date: November 2016

import math

def get\_perimeter\_square(side):

perimeter = side \* 4

return perimeter

def get\_perimeter\_triangle(side):

perimeter = side \* 3

return perimeter

def get\_perimeter\_circle(radius):

perimeter = 2 \* math.pi \* radius

return perimeter

def get\_area\_square(side):

area = side \*\* 2

return area

def get\_area\_triangle(side):

area = (side / 2) \* side

return area

def get\_area\_circle(radius):

area = math.pi \* (radius \*\* 2)

return area

def get\_volume\_cube(side):

volume = side \*\* 3

return volume

def get\_volume\_tetrahedron(side):

volume = (side \*\* 3) / 6 \* math.sqrt(2)

return volume

def get\_volume\_sphere(radius):

volume = 4 / 3 \* math.pi \* (radius \*\* 3)

return volume

user\_input = input("please input square, triangle or circle followed by a comma, then volume, perimeter or area followed by a comma and then either the length of the side or the radius.\n\n")

split\_input = user\_input.split(",")

if split\_input[0] == "square":

if split\_input[1] == "perimeter":

print(get\_perimeter\_square(int(split\_input[2])))

elif split\_input[1] == "area":

print(get\_area\_square(int(split\_input[2])))

else:

print(get\_volume\_cube(int(split\_input[2])))

elif split\_input[0] == "triangle":

if split\_input[1] == "perimeter":

print(get\_perimeter\_triangle(int(split\_input[2])))

elif split\_input[1] == "area":

print(get\_area\_triangle(int(split\_input[2])))

else:

print(get\_volume\_tetrahedron(int(split\_input[2])))

else:

if split\_input[1] == "perimeter":

print(get\_perimeter\_circle(int(split\_input[2])))

elif split\_input[1] == "area":

print(get\_area\_circle(int(split\_input[2])))

else:

print(get\_volume\_sphere(int(split\_input[2])))

'''

#assertions

Input: circle,volume,4

Output: 268.082573106329

Input: triangle,perimeter,6

Output: 18

'''