COMP4004 Problem Solving & Programming

Template for Week 2 and 3 Portfolio Exercises

You should use this file as a template for the Turnitin submission of your Portfolio Exercises for Weeks 2 and 3.

The solutions that you paste in here can be different to the ones originally uploaded to Replit.

Please ONLY upload your solutions to the week 2 and week 3 portfolio exercises. You will upload other pieces of work later.

# Academic Conduct

The contents of your portfolio must be your own, individual work. You should ensure that you are aware of the university rules on plagiarism and collusion.

# Portfolio Exercises

Put your submissions to the portfolio exercises here. Use a fixed width font such as Courier New to preserve the indentation of your code when you paste it in.

N.B. You must upload your submission as a Word document with the extension .doc, or .docx, or as an Open Office document (extension .odt, .doc, or .docx). Do **NOT** upload documents with the extension .py, because they cannot be processed by Turnitin. Do **NOT** upload PDF documents, because these pose problems for us if we try to run your code.

**Note: If you have problems copying and pasting from Replit to Word, send an email to the module leader (**[**dsutton@brookes.ac.uk**](mailto:dsutton@brookes.ac.uk)**). The process can sometimes be a bit temperamental, but I will probably be able to suggest a solution.**

# Week 2

## WK2EX6 [P] Postal Program

if \_\_name\_\_ == "\_\_main\_\_":

weight = int(input('Please insert the weight of the package in grams: '))

length = int(input('Please insert the length of the package in centimeters: '))

diameter = int(input('Please insert the diameter of the package in centimeters: '))

length\_diameter = length + (diameter\*2)

if weight<= 2000 and length <= 90 and length\_diameter <= 104:

print('Yes')

else:

print('No')

## WK2EX7 [P] Overlapping Events (V3)

if \_\_name\_\_ == "\_\_main\_\_":

# A program to detect overlapping events

# User input commands for event A and B

start\_hourA = int(input('Enter the start hour of event A:'))

durationA = int(input('Enter the duration of event A (in hours):'))

end\_hourA = start\_hourA + durationA

start\_hourB = int(input('Enter the start hour of event B:'))

durationB = int(input('Enter the duration of event B (in hours):'))

end\_hourB = start\_hourB + durationB

#Here is the logic to determine the overlapping

if start\_hourB >= end\_hourA and end\_hourB > start\_hourA or start\_hourA >= end\_hourB and end\_hourA > start\_hourB:

print ('Events do not overlap')

else:

print('Events overlap')

# Week 3

## WK3EX4 [P] Cumulative Values

if \_\_name\_\_ == "\_\_main\_\_":

# Creation of lists

list\_one = []

list\_two = []

# User input code

user\_sequence = int(input('Enter an unsigned integer (or -1 to finish): '))

# While loop to gather list\_one

while user\_sequence != -1:

list\_one.append(user\_sequence)

user\_sequence = int(input('Enter an unsigned integer (or -1 to finish): '))

# For loop to calculate cumulative frequency of list one

cumulative\_sum = 0

for number in list\_one:

if number != -1:

cumulative\_sum += number

list\_two.append(cumulative\_sum)

else:

break

# Print second list

for numbers in list\_two:

print(numbers)

## WK3EX5 [P] List of words

if \_\_name\_\_ == "\_\_main\_\_":

# Create lists for test 1 and test 2

commands\_test\_one = ['add add','add find','add cat','add bat','add smart','add tart','find add','find find','find cat','find bat','find smart','find tart']

commands\_test\_two = ['add house','add house','add shed','add hovel','find house','find shed','find hovel']

list\_of\_words = []

list\_of\_words\_two = []

# User input code

user\_input = str(input('Enter your command: '))

# Split input words for test 1 and test 2

for user\_input in commands\_test\_one:

command\_one = user\_input.split()[0]

for user\_input in commands\_test\_two:

command\_two = user\_input.split()[0]

# Responding differently to 'add' and 'find' commands

# Test 1

if command\_one == 'add':

word = user\_input[len('add '):]

list\_of\_words.append(word)

elif command\_one == 'find':

find\_find = user\_input[len('find '):]

find\_found = [word for word in list\_of\_words if find\_find in word]

for found\_find in find\_found:

print(found\_find)

elif command == 'quit':

break

# Test 2

if command\_two == 'add':

word\_2 = user\_input[len('add '):]

list\_of\_words\_two.append(word\_2)

elif command\_two == 'find':

find\_find\_2 = user\_input[len('find '):]

find\_found\_2 = [word\_2 for word\_2 in list\_of\_words\_two if find\_find\_2 in word\_2]

for found\_find\_2 in find\_found\_2:

print(found\_find\_2)

elif command\_two == 'quit':

break