**Course: 22-23: HDip in Computing in Data Analytics**

**Author: Sarah Hastings**

**Subject: Programming and Scripting**

**This file contains the files for the weekly tasks saved to location pands-problems-sheet**

[**https://github.com/Sarahlouhast/pands-problems-sheet.git**](https://github.com/Sarahlouhast/pands-problems-sheet.git)

**Below is a list of each file name and what the program will do once executed.**

***Weekly Task number 1:***

$ hello world.py File 1 - Week 1 task called hello world. This file should contain a python program that displays Hello World! when it is run.

Code:

print("Hello World!")

***Weekly Task number 2:***

$ bank.py File 2 - Week 2 task called bank.py This program should prompt the user and read in two money amounts (in cent) Add the two amounts Print out the answer in a human readable format with a euro sign and decimal point between the euro and cent of the amount For example $ python bank.py Enter amount1(in cent): 65 Enter amount2(in cent): 180 The sum of these is €2.45

Code:

#Prompt the user to enter integer amounts

num1 = int(input("Enter amount1(in cent):"))

num2 = int(input("Enter amount2(in cent):"))

sum= num1 + num2 #Calculate the sum of the values entered

format\_sum = format(sum/100) #Format the value

print (f"The sum of these is €{format\_sum}")

Code Explanation:

References:

<https://www.w3schools.com/python/python_numbers.asp> <https://www.w3schools.com/python/python_howto_add_two_numbers.asp> <https://stackabuse.com/format-number-as-currency-string-in-python/>

***Weekly Task number 3:***

$ accounts.py File 3 - Week 3 task called accounts.py This program that reads in a 10 character account number and outputs the account number with only the last 4 digits showing (and the first 6 digits replaced with Xs). Modify the program to deal with account numbers of any length.

For example $ python accounts.py Please enter an 10 digit account number: 1234567890 XXXXXX7890

Code:

while True:

accountno = input("Please enter an 10 digit account number:")

if len(accountno) !=10:

print('That was not a 10 digit number. Please enter your 10 digit number \n')

else:

Bank = accountno[-4:].rjust(len(accountno), 'X') #When 10 digits are entered, the rjust() method will right align the string using X as the fill character to anything before the last 4 digits

print(Bank)

break

account\_no\_1 = input("Please enter an account number: ")

print("x" \* (len(account\_no\_1) - 4) + account\_no\_1[-4:])

Code Explanation:

This code will take only input of 10 numbers, keep asking until the user enters 10 numbers, print error message if 10 digits are not entered. Extra step to the program will modify the program to deal with account numbers of any length, and assuming the next step is to display only the last 4 characters, all other numbers will be replaced with Xs and the end result printed out.

References:

<https://www.w3schools.com/python/python_strings_slicing.asp>

<https://www.freecodecamp.org/news/python-slicing-how-to-slice-an-array/>

<https://www.w3schools.com/python/ref_string_rjust.asp>

***Weekly Task number 4:***

$ collatz.py File 4 - Week 4 task, called collatz.py, This program asks the user to input any positive integer and outputs the successive values of the following calculation. At each step calculate the next value by taking the current value and, if it is even, divide it by two, but if it is odd, multiply it by three and add one. Have the program end if the current value is one.

For example $ python collatz.py

Please enter a positive integer: 10

10 5 16 8 4 2 1

Code:

#n allows user to enter a number

n = int(input("Please enter a positive integer: "))

while n !=1: #While loop will continue until n is 1

print(n, end= ' ')

if n % 2 == 0: # A number is even if division by 2 gives a remainder of 0

n = n // 2 # To avoid decimal, use floor division with, this will always give a integer value

# https://data-flair.training/blogs/python-operator/

else:

n = n \* 3 + 1 #Else is needed for odd numbers, where remainder is 1, multiply \*3 +1

else:

print(n, end= ' ') #end= ' ' allows you choose what you want to be printed after the print statement has been executed

#https://www.w3schools.com/python/ref\_func\_print.asp

Code Explanation:

References:

***Weekly Task number 5:***

$ weekday.py File 5 - Week 5 task, called weekday.py, This program outputs whether or not today is a weekday. An example of running this program on a Thursday is given below.

$ python weekday.py Yes, unfortunately today is a weekday. An example of running it on a Saturday is as follows: $ python weekday.py It is the weekend, yay!

Code:

from datetime import datetime #Firstly need to import "datetime" to work with dates

weekno = datetime.today().weekday() # Get Day Number from weekday

if weekno < 5:

print("Yes, unfortunately today is a weekday.")

else:

print("It is the weekend, yay!") # 5 Sat, 6 Sun

Code Explanation:

Firstly, we must import module "datetime" to work with dates in Python.

References:

<https://www.w3schools.com/python/python_datetime.asp>

<https://www.programiz.com/python-programming/datetime/current-datetime>

<https://www.datacamp.com/tutorial/python-datetime>

***Weekly Task number 6:***

$ squareroot.py File 6 - Week 6 task, called squareroot.py, This program takes a positive floating-point number as input and outputs an approximation of its square root. This program will use the newton method at estimating square roots.

For example $ python squareroot.py Please enter a positive number: 14.5 The square root of 14.5 is approx. 3.8.

Code:

num = float(input("Please enter a positive number: ")) #User is prompted to enter a positive float number

def newtonsqrt(n): #This function will calculate the squart root using newton's method

approx=0.5\*n #Start with any approximation

better=0.5\*(approx+n/approx) #Compute a better approximation

while better!=approx: #Adding the while condition here to execute until the approximation is no longer changing

approx=better

better=0.5\*(approx+n/approx)

return round(approx,1)

print(f"The square root of {num} is approx. {newtonsqrt(num)}.") #This will call the function and print the output results

Code Explanation:

References:

<https://www.simplilearn.com/tutorials/python-tutorial/float-in-python#:~:text=Float()%20is%20a%20method,as%20the%20floating%2Dpoint%20output>.

<https://thirumalai2024.medium.com/python-program-to-find-square-root-of-the-number-using-newtons-method-937c0e732756>

<https://www.geeksforgeeks.org/find-root-of-a-number-using-newtons-method/>

<https://towardsdatascience.com/newton-raphson-explained-and-visualised-23f63da21bd5> <https://runestone.academy/ns/books/published/thinkcspy/MoreAboutIteration/NewtonsMethod.html>

***Weekly Task number 7:***

$ moby-dick.py File 7 - Week 7 task, called moby-dick.py, This program reads in a text file and outputs the number of e's it contains. The program should take the filename from an argument on the command line.

For example $ python es.py moby-dick.txt 116960

Code:

import sys

filename= sys.argv[1] #sys.argv is a list that contains all the arguments passed to the script on the command line

def letterFrequency(fileName, letter): #This function is created to return the letter count in the text file

try: #Adding the try except incase the filename entered does not exist, https://www.w3schools.com/python/python\_try\_except.asp

file = open(fileName, 'r') #This will open the file in read mode

text = file.read() #This will store the content of the file in a variable

return text.count(letter) #Using the count() function to return the number of times a given value/letter occurs in the file

except FileNotFoundError:

print("Filename (", filename,") does not exist. Please re-run and enter a valid filename.", sep = '')

sys.exit(1)

print(letterFrequency((filename), 'e'), (f"occurences of the letter 'e' in the text file.")) #This will call the function and display the letter count of occurences of lower case'e'

Code Explanation:

References:

<https://www.pythonforbeginners.com/system/python-sys-argv>

<https://stackoverflow.com/questions/18047879/opening-files-with-python>

<https://www.w3schools.com/python/python_try_except.asp> <https://www.geeksforgeeks.org/count-the-number-of-times-a-letter-appears-in-a-text-file-in-python/>

***Weekly Task number 8:***

$ plottask.py File 8 - Week 8 task, called plottask.py, This program displays a histogram of a normal distribution of a 1000 values with a mean of 5 and standard deviation of 2, and a plot of the function h(x)=x3 in the range [0, 10], on the one set of axes.

Code:

Code Explanation:

References: