SEAS 6414

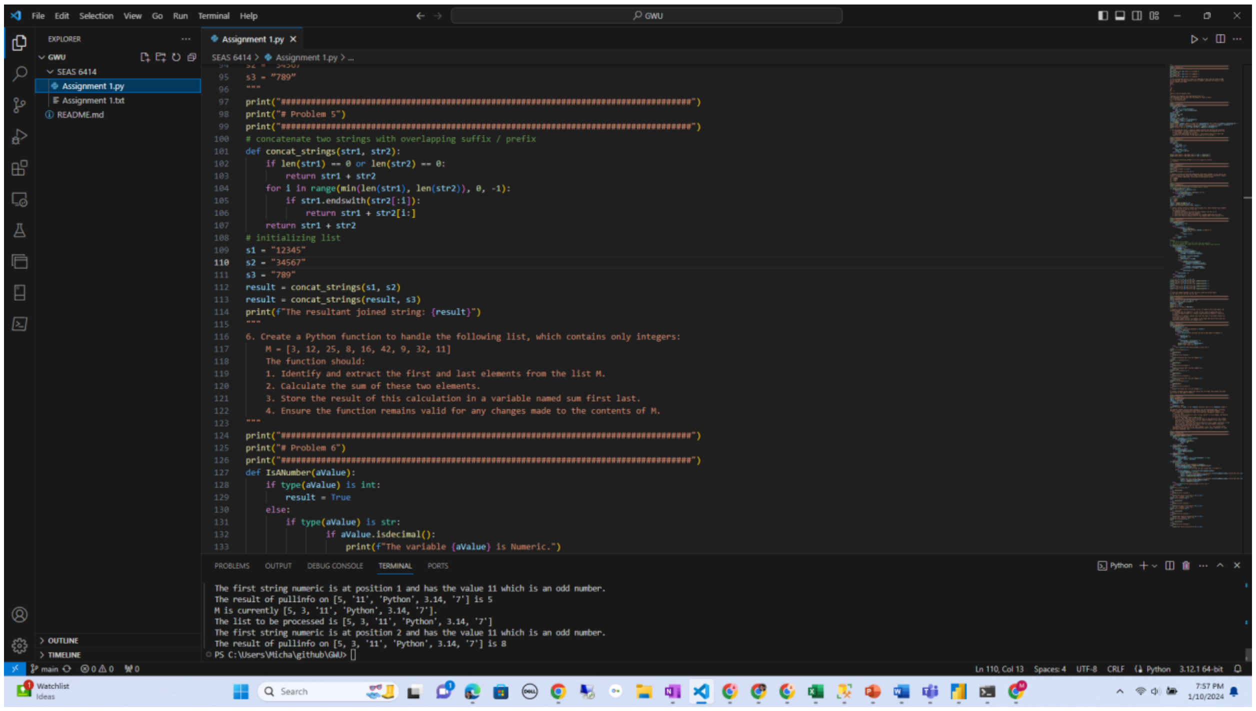
Spring 2024

Dr. Adewale Akinfaderin

I normally use Anaconda for Python but I decided to try Microsoft Visual Studio Code. It worked well and I will keep using it as long as I can. My setup is pictured below. All the assignments will be in GitHub at https://github.com/OwlSaver/GWU. I put the instructions in a text file and then started coding each solution after the instructions. Below you will find my setup, the code, and the output. If this is not convenient for you, et me know and I wil change the format.

My Setup

I downloaded the latest Python distribution from python.org. I already had Microsoft Visual Code installed so I added the python tools. It worked right away. I set up a new project in GitHub and will keep the code there.



```python
s3 = "789"
"""
print("################################################################################")
print("# Problem 5")
print("################################################################################")
# concatenate two strings with overlapping suffix / prefix
def concat_strings(str1, str2):
    if len(str1) == 0 or len(str2) == 0:
        return str1 + str2
    for i in range(min(len(str1), len(str2)), 0, -1):
        if str1.endswith(str2[:i]):
            return str1 + str2[i:]
    return str1 + str2
# initializing list
s1 = "12345"
s2 = "34567"
s3 = "789"
result = concat_strings(s1, s2)
result = concat_strings(result, s3)
print(f"The resultant joined string: {result}")
"""
6. Create a Python function to handle the following list, which contains only integers:
    M = [3, 12, 25, 8, 16, 42, 9, 32, 11]
    The function should:
    1. Identify and extract the first and last elements from the list M.
    2. Calculate the sum of these two elements.
    3. Store the result of this calculation in a variable named sum first last.
    4. Ensure the function remains valid for any changes made to the contents of M.
"""
print("################################################################################")
print("# Problem 6")
print("################################################################################")
def IsANumber(aValue):
    if type(aValue) is int:
        result = True
    else:
        if type(aValue) is str:
                if aValue.isdecimal():
                    print(f"The variable {aValue} is Numeric.")
```

```
The first string numeric is at position 1 and has the value 11 which is an odd number.
The result of pullinfo on [5, '11', 'Python', 3.14, '7'] is 5
M is currently [5, 3, '11', 'Python', 3.14, '7'].
The list to be processed is [5, 3, '11', 'Python', 3.14, '7']
The first string numeric is at position 2 and has the value 11 which is an odd number.
The result of pullinfo on [5, 3, '11', 'Python', 3.14, '7'] is 8
PS C:\Users\Micha\github\GWU>
```

Source Code

Below is the text for each problem followed by the solution. I used the python multiline string capability to delimit the problem text. This is using a fixed space font so the indentation should be correct but there may be some discrepancies, the code is properly indented in the source file.

```
"""
1. Determine the total number of bytes contained in one Gigabyte, given that one
kilobyte equals 1024 bytes, one megabyte equals 1024 kilobytes, and one gigabyte
equals 1024 megabytes.
"""
print("##################################################################################")
print("# Problem 1")
print("##################################################################################")
kb = 1024
print(f"There are {kb} bytes in a kilobyte.")
mb = 1024 * kb
print(f"There are {mb} bytes in a megabyte.")
gb = 1024 * mb
print(f"There are {gb} bytes in a gigabyte.")
"""
2. You purchased 450 shares of stock B at $550 each on day 0 and sold them at $950
each on day 650. The daily discount rate is 0.00015%. Calculate the profit in terms
of net present value (NPV).
Note:
NP V =
X
T
t=0
Rt
(1 + r)
t
• NP V is the net present value.
•
P denotes the summation over time periods from 0 to T.
• Rt represents the net cash flow (the amount of cash) at time t.
• r is the discount rate.
• t is the time period.
"""
print("##################################################################################")
print("# Problem 2")
print("##################################################################################")
def NPVCalc(rate, values):
    pspv = 0
    for idx, cf in enumerate(values):
        pspv = pspv + (cf/((1+rate)**idx))
    return pspv
shares = 450
purchaseprice = 550
saleprice = 950
days = 650
cfvalues = [0] * (days + 1)
cfvalues[0] = shares * -1 * purchaseprice
cfvalues[650] = shares * saleprice
ir = 0.0000015
print(f"Bought {shares} shares on day 0 for ${purchaseprice} for a total of ${shares *
purchaseprice}.")
print(f"Sold {shares} shares on day 650 for ${saleprice} for a total of ${shares * saleprice}.")
print(f"Total profit is ${(shares * saleprice) - (shares * purchaseprice)}.")
print(f"Using a rate of %{ir * 100} the NPV is {NPVCalc(ir,cfvalues)}.")
"""
3. For any numerical value x, develop a Python function that performs the following:
    1. Instantiate a variable named even check which should be assigned True if x is
    an even number, and False if x is odd.
    2. Along with determining the parity of x, the function should also return the
    square of x if it is even, or the square root of x if it is odd.
"""
print("##################################################################################")
print("# Problem 3")
print("##################################################################################")
import math
def esosr(x):
    if x % 2 == 0:
        even_check = True
        retval = x * x
    else:
        even_check = False
        retval = math.sqrt(x)
    return retval

print(f"Even Square / Odd Square Root of {22} is {esosr(22)}")
print(f"Even Square / Odd Square Root of {23} is {esosr(23)}")

"""
4. Correct the following variable (s) so it is equal to ”soccer”:
s = ”sooter”
"""
print("##################################################################################")
print("# Problem 4")
print("##################################################################################")
s = "sooter"
print(f"The variable s is {s}")
s = "soccer"
print(f"The variable s is now {s}.")
"""
5. Develop a Python function that manipulates three given strings, s1, s2, and s3, to
create a single string, consec ints, which contains a sequence of consecutive integers
from 1 to 9 without repetition. The strings are defined as follows:
s1 = ”12345”
s2 = ”34567”
s3 = ”789”
"""
print("##################################################################################")
print("# Problem 5")
print("##################################################################################")
# concatenate two strings with overlapping suffix / prefix
def concat_strings(str1, str2):
    if len(str1) == 0 or len(str2) == 0:
        return str1 + str2
    for i in range(min(len(str1), len(str2)), 0, -1):
        if str1.endswith(str2[:i]):
            return str1 + str2[i:]
    return str1 + str2
# initializing list
s1 = "12345"
s2 = "34567"
s3 = "789"
result = concat_strings(s1, s2)
result = concat_strings(result, s3)
print(f"The resultant joined string: {result}")
"""
6. Create a Python function to handle the following list, which contains only integers:
    M = [3, 12, 25, 8, 16, 42, 9, 32, 11]
    The function should:
    1. Identify and extract the first and last elements from the list M.
    2. Calculate the sum of these two elements.
    3. Store the result of this calculation in a variable named sum first last.
    4. Ensure the function remains valid for any changes made to the contents of M.
"""
print("##################################################################################")
print("# Problem 6")
print("##################################################################################")
def IsANumber(aValue):
    if type(aValue) is int:
        result = True
    else:
        if type(aValue) is str:
                if aValue.isdecimal():
                    print(f"The variable {aValue} is Numeric.")
                    result = True
                else:
                    result = False
        else:
            result = False
    return result
#
# Notes:
#   Only works with integers
#   If list is 1 item long, uses that item, does not double it
#   Any error results in a return of zero, not useful, should raise and error
def SumFirstLast(aList):
    if type(aList) is list:
        firstNumPos = 0
        lastNumPos = len(aList) - 1
        if IsANumber(aList[firstNumPos]):
            if type(aList[firstNumPos]) is str:
                firstNum = int(aList[firstNumPos])
            else:
                firstNum = aList[firstNumPos]
            if len(aList) == 1:
                sumfirstlast = firstNum
            else:
                if IsANumber(aList[lastNumPos]):
                    if type(aList[lastNumPos]) is str:
                        lastNum = int(aList[lastNumPos])
                    else:
                        lastNum = aList[lastNumPos]
                    sumfirstlast = firstNum + lastNum
                else:
                    sumfirstlast = 0
        else:
            sumfirstlast = 0
    else:
        sumfirstlast = 0
    return sumfirstlast

M = [3, 12, 25, 8, 16, 42, 9, 32, 11]
print(f"Case 1: List {M} Sum First Last {SumFirstLast(M)}.")
M = [9, 12, 25, 8, 16, 42, 9, 32, 77]
print(f"Case 2: List {M} Sum First Last {SumFirstLast(M)}.")
M = [3]
print(f"Case 3: List {M} Sum First Last {SumFirstLast(M)}.")
M = ["A", 12, 25, 8, 16, 42, 9, 32, 11]
print(f"Case 4: List {M} Sum First Last {SumFirstLast(M)}.")

"""
7. Slice and combine elements of the list M to create the string ”data”.
M = [4, ”d”, 8, ”t”, 15, ”a”, 16, 23, ”a”]
"""
print("##################################################################################")
print("# Problem 7")
print("##################################################################################")
M = [4, "d", 8, "t", 15, "a", 16, 23, "a"]
N = M[1:2:]+M[5:6:]+M[3:4:]+M[8:9:]
NString = ''.join(N)
print(f"M is {M}")
print(f"N is {N}")
print(f"N as a string is {NString}")
"""
8. Develop a Python function to process a list L of numbers with an odd length. The
function should:
    1. Validate that the length of L is odd. If not, raise an appropriate error.
    2. Find the median element of L without using any built-in median or statistics
    functions.
    3. Slice out all elements indexed lower than the index of the median element and
    store them in a new list.
    You should create the list L of numbers of odd length and ensure these tasks are
    performed without altering the original list L.
"""
print("##################################################################################")
print("# Problem 8")
print("##################################################################################")
def medianFind(aList):
    if type(aList) is list:
        print(f"The list to be processed is {aList}")
        aListLen = len(aList)
        if aListLen % 2 == 0:
            raise Exception("The list does not have an odd number of elements.")
        else:
            if aListLen == 1:
                medianIndex = 0
                LL = []
            else:
                medianIndex = int(aListLen / 2)
                LL = aList[0:medianIndex:1]
            print(f"Median index is {medianIndex}")
            print(f"Lower values are {LL}")
    else:
        raise TypeError("The value provided is not a list.")
L = 17
print(f"L is currently {L}.")
try:
    medianFind(L)
except:
    print("An error occurred.")
else:
    print(f"Verifying that L has not changed {L}")
L = [1]
print(f"L is currently {L}.")
try:
    medianFind(L)
except:
    print("An error occurred.")
else:
    print(f"Verifying that L has not changed {L}")
L = [1,2,3,4]
print(f"L is currently {L}.")
try:
    medianFind(L)
except:
    print("An error occurred.")
else:
    print(f"Verifying that L has not changed {L}")
L = [1,2,3,4,5]
print(f"L is currently {L}.")
try:
    medianFind(L)
except:
    print("An error occurred.")
else:
    print(f"Verifying that L has not changed {L}")
"""
9. Create a variable called ”name” that stores your full name. Find whether your name
has an even or odd number of letters.
"""
print("##################################################################################")
print("# Problem 9")
print("##################################################################################")
name = "Michael Wacey"
nameLen = len(name)
if nameLen % 2 == 0:
    nameCount = "Even"
else:
    nameCount = "Odd"
print(f"My name is {name}, it has {nameLen} characters which is an {nameCount} number.")
"""
10. Develop a Python function that processes a list M of mixed data types, including
integers, strings, and potentially other Python objects. The function should:
    1. Validate that M contains at least one string that represents a numeric value
    (e.g., '3', '42').
    2. Find the first occurrence of such a string, convert it to an integer, and identify
    its position in the list.
    3. Check if the integer value is odd or even:
        • If it's odd, slice the list M from the start to the position of this integer
        (exclusive) and calculate the sum of all numeric elements in this slice. Store
        this sum in a variable odd sum.
        • If it's even, slice the list from the position of this integer (exclusive) to the
        end of the list (inclusive) and create a new list even elements containing
        only string elements from this slice.
    For example, given a list M = [5, ”10”, ”Python”, 3.14, ”7”], the function should
    identify ”10” as the first string representing a numeric value, determine it's even,
    and return [”Python”, ”7”].
"""
print("##################################################################################")
print("# Problem 10")
print("##################################################################################")
def FindFirstStringNumeric(aList):
    result = 0
    for idx, anElement in enumerate(aList):
        if type(anElement) is str:
            if anElement.isdecimal():
                result = idx
                break
    return result
def FindOnlyStrings(aList):
    result = []
    for anElement in aList:
        if type(anElement) is str:
            result.append(anElement)
    return result
def SumNumerics(aList):
    result = 0
    for anElement in aList:
        if type(anElement) is int or type(anElement) is float:
            result = result + anElement
    return result
def pullinfo(aList):
    if type(aList) is list:
        print(f"The list to be processed is {aList}")
        firstStringNumericIndex = FindFirstStringNumeric(aList)
        if firstStringNumericIndex == 0:
            raise Exception("The list does not have string that is a numeric.")
        else:
            firstStringNumeric = int(aList[firstStringNumericIndex])
            if firstStringNumeric % 2 == 0:
                print(f"The first string numeric is at position {firstStringNumericIndex} and has
the value {firstStringNumeric} which is an even number.")
                newList = aList[firstStringNumericIndex+1::]
                result = FindOnlyStrings(newList)
            else:
                print(f"The first string numeric is at position {firstStringNumericIndex} and has
the value {firstStringNumeric} which is an odd number.")
                newList = aList[0:firstStringNumericIndex:]
                result = SumNumerics(newList)
    else:
        raise TypeError("The value provided is not a list.")
    return result
M = 17
print(f"M is currently {M}.")
try:
    R = pullinfo(M)
except:
    print("An error occurred.")
else:
    print(f"The result of pullinfo on {M} is {R}")
M = [5, "10", "Python", 3.14, "7"]
print(f"M is currently {M}.")
try:
    R = pullinfo(M)
```

```
except:
    print("An error occurred.")
else:
    print(f"The result of pullinfo on {M} is {R}")
M = [5, "11", "Python", 3.14, "7"]
print(f"M is currently {M}.")
try:
    R = pullinfo(M)
except:
    print("An error occurred.")
else:
    print(f"The result of pullinfo on {M} is {R}")
M = [5, 3, "11", "Python", 3.14, "7"]
print(f"M is currently {M}.")
try:
    R = pullinfo(M)
except:
    print("An error occurred.")
else:
    print(f"The result of pullinfo on {M} is {R}")
```

Output

Below is the output that I captured using redirection.

```
#################################################################################
# Problem 1
#################################################################################
There are 1024 bytes in a kilobyte.
There are 1048576 bytes in a megabyte.
There are 1073741824 bytes in a gigabyte.
#################################################################################
# Problem 2
#################################################################################
Bought 450 shares on day 0 for $550 for a total of $247500.
Sold 450 shares on day 650 for $950 for a total of $427500.
Total profit is $180000.
Using a rate of %0.00015000000000000001 the NPV is 179583.39094237896.
#################################################################################
# Problem 3
#################################################################################
Even Square / Odd Square Root of 22 is 484
Even Square / Odd Square Root of 23 is 4.795831523312719
#################################################################################
# Problem 4
#################################################################################
The variable s is sooter
The variable s is now soccer.
#################################################################################
# Problem 5
#################################################################################
The resultant joined string: 123456789
#################################################################################
# Problem 6
#################################################################################
Case 1: List [3, 12, 25, 8, 16, 42, 9, 32, 11] Sum First Last 14.
Case 2: List [9, 12, 25, 8, 16, 42, 9, 32, 77] Sum First Last 86.
Case 3: List [3] Sum First Last 3.
Case 4: List ['A', 12, 25, 8, 16, 42, 9, 32, 11] Sum First Last 0.
#################################################################################
# Problem 7
#################################################################################
M is [4, 'd', 8, 't', 15, 'a', 16, 23, 'a']
N is ['d', 'a', 't', 'a']
N as a string is data
#################################################################################
```

```
# Problem 8
##################################################################################
L is currently 17.
An error occurred.
L is currently [1].
The list to be processed is [1]
Median index is 0
Lower values are []
Verifying that L has not changed [1]
L is currently [1, 2, 3, 4].
The list to be processed is [1, 2, 3, 4]
An error occurred.
L is currently [1, 2, 3, 4, 5].
The list to be processed is [1, 2, 3, 4, 5]
Median index is 2
Lower values are [1, 2]
Verifying that L has not changed [1, 2, 3, 4, 5]
##################################################################################
# Problem 9
##################################################################################
My name is Michael Wacey, it has 13 characters which is an Odd number.
##################################################################################
# Problem 10
##################################################################################
M is currently 17.
An error occurred.
M is currently [5, '10', 'Python', 3.14, '7'].
The list to be processed is [5, '10', 'Python', 3.14, '7']
The first string numeric is at position 1 and has the value 10 which is an even number.
The result of pullinfo on [5, '10', 'Python', 3.14, '7'] is ['Python', '7']
M is currently [5, '11', 'Python', 3.14, '7'].
The list to be processed is [5, '11', 'Python', 3.14, '7']
The first string numeric is at position 1 and has the value 11 which is an odd number.
The result of pullinfo on [5, '11', 'Python', 3.14, '7'] is 5
M is currently [5, 3, '11', 'Python', 3.14, '7'].
The list to be processed is [5, 3, '11', 'Python', 3.14, '7']
The first string numeric is at position 2 and has the value 11 which is an odd number.
The result of pullinfo on [5, 3, '11', 'Python', 3.14, '7'] is 8
```