1811ICT/2807ICT/7001ICT Programming Principles Workshop 8

School of Information and Communication Technology

Griffith University

|  |  |
| --- | --- |
| *Goals* | This workshop focusses on everything in the course up to files. |
| When | Workshops from Friday 13 May to Thursday 19 May |
| Marks | 3 |
| Due | There are no pre-workshop tasks this week.  Workshop programming problems by 11:59pm on 22 May |

# Before your workshop class:

* Read all of this document.
* Review the lecture notes sections 1 to 21.
* **There are no pre-workshop tasks this week**.

# Workshop activities

## Problem 1

*Problem:* Write a program with a *function* that given a list of numbers, rotate the numbers so the first number becomes the last, and the rest of the numbers move one position forward. Do the rotation iteratively until the list of numbers returns to its initial form.

Input a list: 1 2 3 4

[1, 2, 3, 4]

[2, 3, 4, 1]

[3, 4, 1, 2]

[4, 1, 2, 3]

[1, 2, 3, 4]

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following two scenarios:

* 4, 2, 4, 3, 3, 1, 2
* 1, 2, 1, 2

***Copy your code here***

# Matthew Prendergast

# 17th May, 2022 - Problem 1 (Workshop - Week 8)

# Define the rotation function.

def rotate(numbers):

    """ The function will take in a list of numbers and rearrange it until it is back to its original"""

    temp = []

    for i in range(len(numbers)):

        for j in range(1, len(numbers)):

            temp.append(numbers[j])

        temp.append(numbers[0])

        print(temp)

        numbers = temp

        temp = []

# Prompt user to input a a list of numbers.

ls = input("Input a list: ")

# Convert the input to a list.

num\_list = ls.split(" ")

for i in range(len(num\_list)):

    num\_list[i] = int(num\_list[i])

print(num\_list)

rotate(num\_list)

***Insert your screenshots here***

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

## Problem 2

*Problem:* Given two lists, write a program with a *function* that takes these two lists as the input arguments and returns the list of all the elements in the first list that occur in the second list. The returned list shall not contain duplicate elements. Your main program will allow the user to enter two lists of numbers and end the program by inputting a blank line for list 1.

|  |
| --- |
| List 1: 1 3 3 6  List 2: 3 4 2 1 2 1 3  Output: [1, 3]  List 1: 3 4 2 1 2 1 3  List 2: 5 6 7 8  Output: []  List 1: |

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following two scenarios:

* List 1: 0 1 2 3 1 2 3 2 3 3

List 2: 2 4 6 1 3 5

* List 1: 1 1 2 2 3 3 4 4

List 2: 8 7 6 5 4 3 2 1 0

***Copy your code here***

# Matthew Prendergast

# 17th May, 2022 - Problem 2 (Workshop - Week 8)

# Define the sorting function.

def sort\_list(x, y):

    sorted = []

    i, j = 0, 0

    while i < len(x) and j < len(y):

        if x[i] == y[j]:

            if x[i] not in sorted:

                sorted.append(x[i])

            i += 1

            j += 1

        elif x[i] < y[j]:

            i += 1

        else:

            j += 1

    print(sorted)

# Prompt user to input a a list of numbers.

l\_one = input("List 1: ")

while l\_one:

    l\_two = input("List 2: ")

    # Convert the inputs to lists.

    l\_one = l\_one.split(" ")

    l\_two = l\_two.split(" ")

    # Convert to the lists to integer lists.

    for i in range(len(l\_one)):

        l\_one[i] = int(l\_one[i])

    for i in range(len(l\_two)):

        l\_two[i] = int(l\_two[i])

    # Sort the lists.

    l\_one.sort()

    l\_two.sort()

    sort\_list(l\_one, l\_two)

    l\_one = input("List 1: ")

***Insert your screenshots here***

Text

Description automatically generated

Text

Description automatically generated

## Problem 3

*Problem:* Write a program with a *function* that given a list of numbers, returns True if and only if all of the numbers in the list form an arithmetic progression, that is the difference between any two successive numbers in the list is the same. Your main program should read a file containing space-separated numbers as test lists, print each list and the outcome after calling the function.

File name: numbers.txt

[1 2 3 4] True

[10 20 30 40] True

[10 9 8 7] True

[2 7 8 3] False

[1 2 3 5] False

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following scenario:

* Use the file P3\_v1.txt as the source file.

***Copy your code here***

# Matthew Prendergast

# 17th May, 2022 - Problem 3 (Workshop - Week 8)

def num\_check(num\_list):

    """This function will calculate the difference between the first two numbers,

       then compare that to the difference between the other numbers in the array"""

    difference = num\_list[0] - num\_list[1]

    for j in range(1, len(num\_list) - 1):

        if num\_list[j] - num\_list[j + 1] != difference:

            return False

    return True

# Prompt user to input a the file names.

file\_read = input("File name: ")

# Open the file to read:

while True:

    try:

        f\_read = open(file\_read, mode = "r")

        break

    except:

        print("Error: File cannot be opened")

        file\_read = input("File name: ")

# For each line in the text file

for line in f\_read:

    # Read the line and convert it to a list. Convert the string elements to integers.

    num\_list = line.split(" ")

    for i in range(len(num\_list)):

        num\_list[i] = int(num\_list[i])

    # Print the list, and the return value of the function to compare the numbers.

    if line[-1] == "\n":

        print(f"[{line[:-1]}]", end=" ")

    else:

        print(f"[{line}]", end=" ")

    print(num\_check(num\_list))

# Close the file.

f\_read.close()

***Insert your screenshots here***

Text

Description automatically generated

## Problem 4

*Problem:* Given two lists, write a program with a *function* that merges these two lists in descending order. Your main program will allow the user to enter two lists of numbers and end the program by inputting a blank line for list 1. You are not allowed to concatenate the two lists into a new list and then call the built-in sort() function to sort the new list in descending order. But you are allowed to sort the two lists in descending order before merge. Don’t worry about the complexity of the merging algorithm.

List 1: 1 3 3 6

List 2: 1 4 5

[6, 5, 4, 3, 3, 1, 1]

List 1: 100

List 2: 1 1 3

[100, 3, 1, 1]

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following two scenarios:

* List 1: 0 1 2 3 1 2 3

List 2: 2 4 6

* List 1: 1 1 2 2 3 3 4 4

List 2: 8 7 6 5 4 3 2 1 0

***Copy your code here***

# Matthew Prendergast

# 18th May, 2022 - Problem 4 (Workshop - Week 8)

def sort\_list(x, y):

    # Sort the lists.

    x.sort(reverse=True)

    y.sort(reverse=True)

    # Merge the lists into one.

    i, j = 0, 0

    new\_list = []

    while i < len(x) and j < len(y):

        if x[i] == y[j]:

            new\_list.append(x[i])

            new\_list.append(y[j])

            i += 1

            j += 1

        elif x[i] > y[j]:

            new\_list.append(x[i])

            i += 1

        else:

            new\_list.append(y[j])

            j += 1

        if i < len(x) and j >= len(y):

            for g in range(i, len(x)):

                new\_list.append(x[g])

        elif j < len(y) and i >= len(x):

            for h in range(j, len(y)):

                new\_list.append(y[h])

    return new\_list

# Prompt user to input a a list of numbers.

l\_one = input("List 1: ")

while l\_one:

    l\_two = input("List 2: ")

    # Convert the inputs to lists.

    l\_one = l\_one.split(" ")

    l\_two = l\_two.split(" ")

    # Convert to the lists to integer lists.

    for i in range(len(l\_one)):

        l\_one[i] = int(l\_one[i])

    for i in range(len(l\_two)):

        l\_two[i] = int(l\_two[i])

    print(sort\_list(l\_one, l\_two))

    l\_one = input("List 1: ")

***Insert your screenshots here***

Text

Description automatically generated

Text

Description automatically generated

## Problem 5

*Problem:* Book club members meet regularly for coffee at a **round cafe table**. Only a few members turn up randomly each time. It is a good thing that they keep records of who comes to each meeting, because one of them has been diagnosed with contagious book fever. It is very likely that if you sit next to someone with book fever, you will catch book fever.

The file meetings1.txt contains the first names of all the attendees at each meeting, in the order they **sat around the table**. It contains:

1. Chuck Trevor
2. Zack Olive Xander Ephraim
3. Ralph Wendy Ephraim Grace Leslie Phil Kathy
4. Binh Harry Ralph Xander Zack Chuck Uma Suzy Phil Kathy
5. Neville Leslie Kathy
6. Neville Harry\* Binh Vince Xander Zack Quisha Olive Phil
7. Yvonne Uma Trevor Fran Olive Phil Kathy
8. Harry Ralph Ephraim Denise Quisha Grace Phil
9. Binh Mandy Xander Ephraim Leslie Olive Fran
10. John Olive Chuck Mandy

Poor Harry, at meeting number 6, is marked with an asterisk because we know he was infected with book fever then.

Write a program that reports the **names and number of club members infected** up to each meeting from the first meeting where anyone was infected, like this:

|  |
| --- |
| Enter file name: meetings1.txt   1. Harry Neville Binh 3 2. Harry Neville Binh 3 3. Harry Neville Binh Ralph Phil 5 4. Harry Neville Binh Ralph Phil Mandy Fran 7 5. Harry Neville Binh Ralph Phil Mandy Fran John Chuck 9 |

The program should work in general for any such file with exactly one name marked with an asterisk. The file meetings2.txt should produce this output.

1. Phil Andrew Kathy 3
2. Phil Andrew Kathy 3
3. Phil Andrew Kathy Ephraim Harry 5
4. Phil Andrew Kathy Ephraim Harry 5
5. Phil Andrew Kathy Ephraim Harry 5
6. Phil Andrew Kathy Ephraim Harry 5
7. Phil Andrew Kathy Ephraim Harry Suzy Leslie 7

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following scenario:

* Use the file P5\_v1.txt as the source file.

***Copy your code here***

# Matthew Prendergast

# 18th May, 2022 - Problem 5 (Workshop - Week 8)

# Define a function to print each result.

def print\_line(ls, infected):

    print(f"{ls[0]:<5}", end=" ")

    for name in infected:

        print(name, end=" ")

    print(len(infected))

# Prompt the user to input the file name.

file\_name = input("Enter file name: ")

# Try to open the file.

while True:

    try:

        f\_read = open(file\_name, mode="r")

        break

    except:

        print("Error: Could not open the file.")

        file\_name = input("Enter file name: ")

begin = False

infected = []

# Read each line of the file.

for line in f\_read:

    ls = []

    temp\_infected = []

    # If the \* for the first infected person has been found, begin the process of finding the rest.

    if begin:

        # Read in a line without the \n character and create a new list.

        ls = line[:-1].split(" ")

        # Create a new list of names without the week number.

        names = ls[1:]

        # Create a temporary infected list, so it keeps the original checking list the same when new names are added.

        temp\_infected += infected

        for name in temp\_infected:

            # If the name of an infected person is at the table, add those to the left and right to the

            # original infected list.

            if name in names:

                count = names.index(name)

                # If the person is at the end of the list, the person to their right will be at the start of the list.

                # Else, add the person to the left and right.

                if count == len(names) - 1:

                    if names[0] not in temp\_infected:

                        infected.append(names[0])

                elif names[count + 1] not in temp\_infected:

                    infected.append(names[count + 1])

                if names[count - 1] not in temp\_infected:

                    infected.append(names[count - 1])

        print\_line(ls, infected)

    # If the first infected person hasn't been found yet.

    if not begin:

        # To find the first infected person we will look for the asterisk in the line.

        if "\*" in line:

            # Once found, flag the being as True to begin checking all rounds after this itteration.

            begin = True

            # Read in a line without the \n character and create a new list.

            ls = line[:-1].split(" ")

            # Create a new list of names without the week number.

            names = ls[1:]

            count = 0

            # Check to find the exact name that is infected, who has an asterisk against their name.

            for name in names:

                if "\*" in name:

                    # Once found, add the person to the infected list, and the people to their left and right.

                    infected.append(name[:-1])

                    infected.append(names[count - 1])

                    # If the person is at the end of the list, the person to their right will be at the start of the list.

                    if count == len(names) - 1:

                        infected.append(names[0])

                        break

                    else:

                        infected.append(names[count + 1])

                        break

                count += 1

            print\_line(ls, infected)

***Insert your screenshots here***

Text

Description automatically generated

# Submission and marking

There are no pre-workshop tasks this week.

For workshop tasks, please submit this document with copied codes and inserted screenshots using the provided submission link in the course website. Students get 3 marks if they complete four or more problems correctly, or 2 marks if they complete three problems correctly, or 1 mark if they complete one or two problems correctly, or 0 marks without any attempt.