DADSA – Justification

Data Structures used:

**Lists method:**

Pythons standard methods such as lists are a suitable approach that make use of its built-in methods that can be of great use in programs and is often referred to as an array. Lists are known to be very versatile in the sense that it can develop in size and is easy modifiable meaning you’re able to change the data with inside the list. Using the list method, I have created a variety of empty list arrays to store crucial parts of data. For example, the main five I started with was a stock list for the stores, a shopping list for the houses and individual empty arrays for each shop (A, B and C).

the first two list arrays were used to store the data from both file A and file B of the week 1 CSV data files given, by doing so I was then able to call on that data by using various methods which including appending and indexing the data in order to gain what was needed. The list method allowed me to use more simpler sorting methods (instead of using classes) when it came to thinking about how I could display the data that was required for the purpose of this program. Using a sorting method such as the greater than method enabled an easier approach for when it came to sorting objects near the end of task 1.

**Why I believe these methods to be a suitable approach:**

Lists enable me to keep attributed data which again enables me to condense and structure my code. By doing this I was able to perform or use a similar method for the multiple values used; I found using lists to be easy, but the overall understanding needed to be improved in order to successfully use them to its full potential. Storing objects within a python list showed its potential and usefulness for building a structure of data that was implemented to successfully call on each function and method used for example I was able to build new columns and index them to show the stores and the items from both CSV files.

When essentially starting this coursework my first idea was to use the class method however I ended up not using classes for this coursework as I found the list method a much easier approach for when it came to obtaining and extracting the data from each CSV file, I was able to do this by using the def function to create multiple functions that enabled me to store those crucial pieces of data withinside that function and then to be called upon that function when necessary.

**Pseudocode for implementation of task 1:**

PSEUDOCODE: TASK 1

IMPORT CSV

EMPTY ARRAY LIST\_STOCK

EMPTY ARRAY LIST\_SHOP

ARRAY LIST\_SHOP\_NAMES

EMPTY ARRAY\_LIST\_HOUSE\_ORDERS

ARRAY LIST\_HOUSESW1

ARRAY LIST\_HOUSESW2

EMPTY ARRAY\_LIST\_WEEK1

EMPTY ARRAY\_LIST\_WEEK2

EMPTY ARRAY\_LIST\_AB\_COMBO

EMPTY ARRAY LIST\_AC\_COMBO

EMPTY ARRAY LIST\_BC\_COMBO

EMPTY ARRAY LIST\_SHOP\_AB

EMPTY ARRAY LIST\_SHOP\_AC

EMPTY ARRAY LIST\_SHOP\_BC

LIST\_DAYS

#PRINT STATE FOR USER INPUT

Print(input(REQUIRES USER TO PRESS THE INPUT BUTTON TO VIEW STOCK AND WEEK LIST)

#FUNCTION WEEK STOCK

FUNCTION Week\_stock ():

OPEN DADSA CWK SHOPPING DATA WEEK 1 FILE A.CSV

READ CSV FILE

GLOBAL LIST\_STOCK

LIST\_STOCK = LIST

FOR COLUMN IN RANGE(LOOP THROUGH RANGES 1, 28)

# APPEND EACH COLUMN TO SPLIT UP DATA FOR EACH STORE

LIST\_STOCK[INDEX COLUMN][ INDEX 3] = ‘Y’

SHOP\_NAMES APPENDS LIST\_STOCK [INDEX COLUMN][INDEX 0] + LIST\_STOCK[INDEX COLUMN][INDEX 1]))

LIST\_STOCK[INDEX COLUMN][INDEX 4] = ‘Y’

SHOP\_NAMES APPENDS LIST\_STOCK [INDEX COLUMN][INDEX 0] + LIST\_STOCK[INDEX COLUMN][INDEX 1]))

LIST\_STOCK[INDEX COLUMN][INDEX 5] = ‘Y’

SHOP\_NAMES APPENDS LIST\_STOCK [INDEX COLUMN][INDEX 0] + LIST\_STOCK[INDEX COLUMN][INDEX 1]))

OPEN DADSA CWK SHOPPING DATA WEEK 1 FILE B.CSV

READ CSV FILE

LIST\_SHOP

RETURN LIST\_SHOP

LIST\_SHOP = FUNCTION NAME

END FUNCTION

#FUNCTION WEEK1

FUNCTION Week\_list():

FOR i IN RANGE(LOOP THROUGH RANGES 2, 29):

LIST\_VARIABLE = str(LIST\_SHOP[INDEX i][INDEX 0]) + str(LIST\_SHOP[INDEX i]index 8:])

LIST\_WEEK1 APPENDS (LIST\_SHOP[INDEX i][INDEX 0:8])

LIST\_WEEK2 APPENDS (LIST\_VARIABLE)

CALL FUNCTION NAME

END FUNCTION

PRINT(“[INDIVIDUAL SHOP NAMES]”)

PRINT(\*LIST\_SHOP\_NAMES, NEWLINE)

PRINT(NEWLINE)

PRINT(“[WEEK 1]”)

PRINT(\*LIST\_WEEK1, NEWLINE)

PRINT(NEWLINE)

PRINT(“[WEEK 2]”)

PRINT(\*LIST\_WEEK2, NEWLINE)

PRINT(NEWLINE)

#PRINT STATE FOR USER INPUT

Print(input(REQUIRES USER TO PRESS THE INPUT BUTTON TO VIEW SHOPPNG SCHEDULE FOR EACH HOUSEHOLD WITHIN WEEK 1 AND WEEK 2)

###########################################################################

#FUNCTION READ HOUSE DATA

FUNCTION read\_house\_data():

FOR i IN RANGE(LOOP THROUGH RANGES 1, 8):

PRINT(“HOUSE”, LIST\_SHOP[INDEX 0][INDEX i]

FOR j IN RANGE(INDEX RANGE OF CSV FILE B):

IF LIST\_SHOP [INDEX j][INDEX i] >= ‘1’:

PRINT(LIST\_STOCK[INDEX j-1][INDEX 1])

PRINT(“”)

FOR x IN RANGE(LOOP THROUGH RANGES 8, 15):

PRINT(“HOUSE”, LIST\_SHOP[INDEX 0][INDEX i])

FOR k IN RANGE(INDEX RANGE OF CSV FILE B):

IF LIST\_SHOP[INDEX k][INDEX x] >= ‘1’:

PRINT(LIST\_STOCK[INDEX k-1][INDEX 1])

PRINT(“”)

#FUNCTION AVAILABLE ITEMS WEEK 1

FUNCTION available\_itemsW1(shop):

FOR i IN RANGE(LOOP THROUGH RANGES 1, 8):

BEGIN COUNT FROM 0

PRINT(“HOUSE”, LIST\_SHOP[INDEX 0][INDEX i]

FOR j IN RANGE(LOOP THROUGH RANGES 1, 29):

IF LIST\_SHOP[INDEX j][INDEX i] >= ‘1’:

FOR ITEM IN SHOP\_LIST:

IF ITEM == LIST\_SHOP[INDEX j][INDEX 0]:

PRINT(ITEM QUANTITY, ITEM AND ITEM PRICE )

INCREMENT COUNT BY 1

PRINT(“TOTAL NUMBER OF REQUIRED ITEMS: “ + str(COUNT))

PRINT(“”)

#FUNCTION AVAILABLE ITEMS WEEK 2

FUNCTION available\_itemsW2(shop):

FOR x IN RANGE(LOOP THROUGH RANGES 8, 15):

BEGIN COUNT FROM 0

PRINT(“HOUSE”, LIST\_SHOP[INDEX 0][INDEX x])

FOR k IN RANGE(LOOP THROUGH RANGES 1, 29):

IF LIST\_SHOP[INDEX k][INDEX x] >= ‘1’:

FOR ITEM IN SHOP\_LIST:

IF ITEM == LIST\_SHOP[INDEX k][INDEX 0]:

PRINT(ITEM QUANTITY, ITEM AND ITEM PRICE )

INCREMENT COUNT BY 1

PRINT(“TOTAL NUMBER OF REQUIRED ITEMS: “ + str(COUNT))

PRINT(“”)

END FUNCTION

#FUNCTION REMOVE DUPLICATED ITEMS

FUNCTION remove\_dup\_items(org\_stores, new\_stores):

FOR DUPLICATE ITEMS in ORIGINAL STORES:

IF DUPLICATE ITEMS not in NEW STORES:

NEW STORE APPENDS DUPLICATE ITEMS

CALL FUNCTION - REMOVE DUPLICATED ITEMS(STORE COMBINATION, STORES COMBINED)

END FUNCTION

PRINT(“\_”\*20, “SHOPPING LIST – WEEK 1 – SHOP\_NAMES”, “\_”\*20)

PRINT(NEWLINE)

CALL FUNCTION - AVAILALE ITEMS WEEK 1(SHOP\_NAMES)

PRINT(“\_”\*20, “SHOPPING LIST – WEEK 2 – SHOP\_NAMES”, “\_”\*20)

PRINT(NEWLINE)

CALL FUNCTION - AVAILALE ITEMS WEEK 2(SHOP\_NAMES)

###########################################################################

#PRINT STATE FOR USER INPUT

Print(input(REQUIRES USER TO PRESS THE INPUT BUTTON TO VIEW DELIVERY SCHEDULE FOR WEEK 1)PRINT(“\_”\*20, “DELIVERY SCHEDULE FOR HOUSES – WEEK 1”, “-“\*20)

#FUNCTION DELIVERY SCHEDULE

FUNCTION delivery\_scheduleW1():

MAX FROM SHOP A = 0

MAX FROM SHOP B = 0

MAX FROM SHOP C = 0

FOR ROWS IN HOUSE ORDERS:

IF(HOUSESW1 AND COLUMNS != ‘PRODUCTS’):

MAX FROM SHOP A = MAX FROM SHOP A + LIST\_SHOP [INDEX COLUMNS][INDEX 0]

MAX FROM SHOP B = MAX FROM SHOP B + LIST\_SHOP [INDEX COLUMNS][INDEX 1]

MAX FROM SHOP C = MAX FROM SHOP C + LIST\_SHOP [INDEX COLUMNS][INDEX 2]

FOR ROWS IN HOUSESW1:

SHOP A = LIST\_SHOP[INDEX 0]

SHOP B = LIST\_SHOP[INDEX 1]

SHOP C = LIST\_SHOP[INDEX 2]

IF(SHOP A GREATER THAN SHOP B AND SHOP A GREATER THAN SHOP C):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, COLUMNS[INDEX 0], ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP B GREATER THAN SHOP C):

IF SHOP C EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

IF(SHOP B GREATER THAN SHOP A AND SHOP B GREATER THAN SHOP C):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, COLUMNS[INDEX 0], ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP A GREATER THAN SHOP C):

IF SHOP C EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”

IF(SHOP C GREATER THAN SHOP A AND SHOP C GREATER THAN SHOP B):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, COLUMNS[INDEX 0] , ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP A GREATER THAN SHOP B):

IF (SHOP B EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

IF(SHOP B GREATER THAN SHOP A):

IF(SHOP A EQUAL TO 0):

PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

ELSE PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

#NO DELIVERY DAY FOR THE DAYS SHOWN IN CODE

PRINT(“DELIVERY DAY: [SPECIFY NO DELIVERY DAYS]”)

#REST DAY FOR DAYS SHOWN IN CODE

PRINT(“DELIVERY DAY: [REST DAY”)

PRINT(“”)

CALL FUNCTION - DELIVER SHCEDULE

PRINT(NEWLINE)

#PRINT STATE FOR USER INPUT

Print(input(REQUIRES USER TO PRESS THE INPUT BUTTON TO VIEW DELIVERY SCHEDULE FOR WEEK 2)

PRINT(“\_”\*20, “DELIVERY SCHEDULE FOR HOUSES – WEEK 2”, “-“\*20)

#FUNCTION DELIVERY SCHEDULE

FUNCTION delivery\_scheduleW2():

MAX FROM SHOP A = 0

MAX FROM SHOP B = 0

MAX FROM SHOP C = 0

FOR ROWS IN HOUSE ORDERS:

IF(HOUSESW1 AND COLUMNS != ‘PRODUCTS’):

MAX FROM SHOP A = MAX FROM SHOP A + LIST\_SHOP [INDEX COLUMNS][INDEX 0]

MAX FROM SHOP B = MAX FROM SHOP B + LIST\_SHOP [INDEX COLUMNS][INDEX 1]

MAX FROM SHOP C = MAX FROM SHOP C + LIST\_SHOP [INDEX COLUMNS][INDEX 2]

FOR ROWS IN HOUSESW2:

SHOP A = LIST\_SHOP[INDEX 0]

SHOP B = LIST\_SHOP[INDEX 1]

SHOP C = LIST\_SHOP[INDEX 2]

IF(SHOP A GREATER THAN SHOP B AND SHOP A GREATER THAN SHOP C):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, COLUMNS[INDEX 0], ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP B GREATER THAN SHOP C):

IF SHOP C EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

IF(SHOP B GREATER THAN SHOP A AND SHOP B GREATER THAN SHOP C):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, COLUMNS[INDEX 0] , ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP A GREATER THAN SHOP C):

IF SHOP C EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”

IF(SHOP B GREATER THAN SHOP A OR SHOP A GREATER THAN SHOP B):

PRINT(“DELIVERY SCHDULE FOR HOUSE:”, ROWS, ‘: DAY FOR DELIVERY [FROM SPECIFIED SHOP])

IF(SHOP A GREATER THAN SHOP B):

IF (SHOP B EQUAL TO 0):

PRINT(“DAY FOR DELIVERY: [FROM SEPCIFIED SHOP]”)

ELSE: PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

IF(SHOP B GREATER THAN SHOP A):

IF(SHOP A EQUAL TO 0):

PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

ELSE PRINT(“DELIVERY DAY: [FROM SPECIFIED SHOP]”)

#NO DELIVERY DAY FOR THE DAYS SHOWN IN CODE

PRINT(“DELIVERY DAY: [ SPECIFY NO DELIVERY DAYS]”)

#REST DAY FOR DAYS SHOWN IN CODE

PRINT(“DELIVERY DAY: [ SPECIFY REST DAYS”)

PRINT(“”)

CALL FUNCTION - DELIVER SHCEDULE

END FUNCTION