Final Project

Food Ordering App

In [18]:

```
print("\n" * 5)                     #Starting after 5x empty lines.

import datetime                       #Deltatime library, to get Real Date information.
import os                             #OS (Operating system) , To provide cross-platform compatibility

list_foods = &#91;]                        #Variable List of foods, names + prices.
list_drinks = &#91;]                       #Variable List of drinks, names + prices.
list_services = &#91;]                     #Variable List of other services, names + prices.

list_item_price = &#91;0] * 100             #Variable List of item prices. Index: 0-39 for foods, index: 40-79 for drinks,
                                      #Index: 80-99 for other services.
var_discount_1 = 200                         #First discount starts.
var_discount_2 = 1000                        #Second discount starts.
var_discount_3 = 5000                        #Third discount starts.
var_discount_1_rate = 0.05                   #First discount rate.
var_discount_2_rate = 0.10                   #Second discount rate.
var_discount_3_rate = 0.15                   #Third discount rate.


navigator_symbol = "/" # This will make the program runnable on any unix based enviroument because it has differnet file system
if os.name == "nt":
    navigator_symbol = "\\" # This will make the program runnable on Windows


def def_default():
    global list_drinks, list_foods, list_services, list_item_order, list_item_price
    list_item_order = &#91;0] * 100                     #Create a list, length 100. Max index number is 99.
def_default()                                       #Index: 0-39 for foods, index: 40-79 for drinks,
                                                    #Index: 80-99 for other services. Global variables.



def def_main():
    while True:                                           #Repeat Menu until stops.
        print("*" * 31 + "MAIN MENU" + "*" * 32 + "\n"     #Design for Main Menu.
              "\t(O) ORDER\n"                              #"*" * 31 means, write (*) 31 times.
              "\t(R) REPORT\n"
              "\t(P) PAYMENT\n"
              "\t(E) EXIT\n" +
              "_" * 72)

        input_1 = str(input("Please Select Your Operation: ")).upper()    #Input, have to choose operation. Make everything UPPER symbol.
        if (len(input_1) == 1):                                           #Checking input length.
            if (input_1 == 'O'):                                          #If input is "O".
                print("\n" * 10)                                         #Create 100 empty lines.
                def_order_menu()                                          #Start Order Menu function.
                break                                                     #Stop repeating Main Menu.
            elif (input_1 == 'R'):                                        #If input is "R".
                print("\n" * 10)                                         #Create 100 empty lines.
                def_report()                                              #Start Report function.
                break                                                     #Stop repeating Main Menu.
            elif (input_1 == 'P'):                                        #If input is "P".
                print("\n" * 10)                                         #Create 100 empty lines.
                def_payment()                                             #Start Payment function.
                break                                                     #Stop repeating Main Menu.
            elif (input_1 == 'E'):                                        #If input is "E".
                print("*" * 32 + "THANK YOU" + "*" * 31 + "\n")           #Good bye comment.
                break                                                     #Stop repeating Main Menu.
            else:                                                                     #If O, R, P, E not inserted then...
                print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")     #Invalid input.
        else:                                                                         #If input length not equal to 1...
            print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")         #Invalid input.

def def_order_menu():                                                                  #yousef
    while True:                                           # While looping to keep menu alive
        print("*" * 31 + "ORDER PAGE" + "*" * 31 + "\n"     # Mail Menu
              "\t(F) FOODS AND DRINKS\n"
              "\t(O) OTHER SERVICES\n"
              "\t(M) MAIN MENU\n"
              "\t(E) EXIT\n" +
              "_" * 72)

        input_1 = str(input("Please Select Your Operation: ")).upper() # Options Handling : F-O-M-E.
        if len(input_1) == 1:
            if (input_1 == 'F'):  #Easy Access Checking Logic
                print("\n" * 10)
                def_food_drink_order() # Show Food/Drinks Menu
                break
            elif (input_1 == 'O'):
                print("\n" * 10)
                def_other_services() # Show Services Menu
                break
            elif (input_1 == 'M'):
                print("\n" * 10)
                def_main() # Show Main Menu
                break
            elif (input_1 == 'E'):
                print("*" * 32 + "THANK YOU" + "*" * 31 + "\n")
                break
            else:
                print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!") # Handling Bad Inputs
        else:
            print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")

def def_full_file_reader():                                                            #mustafa
    file_foods = open('files'+navigator_symbol+'list_foods.fsd', 'r') # Reading Food List
    for i in file_foods: # Line by line reading
        list_foods.append(str(i.strip())) # Adding each line (Food) into an array after applying Strip function to remove out extra spaces in front and back
    file_foods.close()

    file_drinks = open('files'+navigator_symbol+'list_drinks.fsd', 'r') # Reading Drinks List
    for i in file_drinks:
        list_drinks.append(str(i.strip()))
    file_drinks.close()

    file_services = open('files'+navigator_symbol+'list_services.fsd', 'r') # Reading Services
    for i in file_services:
        list_services.append(str(i.strip()))
    file_services.close()

    i = 0
    while i &lt;= (len(list_foods) - 1): #Enumarte through food list to filter out prices and setup print Formatting by replacing spaces with count difference
        if 'RM' in list_foods&#91;i]:
            list_foods&#91;i] = str(list_foods&#91;i]&#91;:list_foods&#91;i].index('RM') - 1]) + ' ' * (20 - (list_foods&#91;i].index('RM') - 1)) + str(list_f
        i += 1

    i = 0
    while i &lt;= (len(list_drinks) - 1):
        if 'RM' in list_drinks&#91;i]:
            list_drinks&#91;i] = str(list_drinks&#91;i]&#91;:list_drinks&#91;i].index('RM') - 1]) + ' ' * (20 - (list_drinks&#91;i].index('RM') - 1)) + str(li
        i += 1

    i = 0
    while i &lt;= (len(list_services) - 1):
        if 'RM' in list_services&#91;i]:
            list_services&#91;i] = str(list_services&#91;i]&#91;:list_services&#91;i].index('RM') - 1]) + ' ' * (20 - (list_services&#91;i].index('RM') - 1))
        i += 1
def_full_file_reader()

def def_file_sorter(): # Applying Sorting to the array to be sorted from A-Z ASC ((AND)) Extracting out prices after sorting and appending them to a prices ar
    global list_foods, list_drinks, list_services
    list_foods = sorted(list_foods)
    list_drinks = sorted(list_drinks)
    list_services = sorted(list_services)

    i = 0
    while i &lt; len(list_foods):
        list_item_price&#91;i] = float(list_foods&#91;i]&#91;int(list_foods&#91;i].index("RM") + 3):]) # Extracting Out "RM" + &#91;SPACE] from and cast out t
        i += 1

    i = 0
    while i &lt; len(list_drinks):
        list_item_price&#91;40 + i] = float(list_drinks&#91;i]&#91;int(list_drinks&#91;i].index("RM") + 3):]) # Applying extraction on 40 and above items whic
        i += 1

    i = 0
    while i &lt; len(list_services):
        list_item_price&#91;80 + i] = float(list_services&#91;i]&#91;int(list_services&#91;i].index("RM") + 3):]) # Applying extraction on 80 and above items
        i += 1
def_file_sorter()

def def_food_drink_order():
    while True:
            print("*" * 26 + "ORDER FOODS &amp; DRINKS" + "*" * 26)
            print(" |NO| |FOOD NAME|         |PRICE|   |  |NO| |DRINK NAME|        |PRICE|")

            i = 0
            while i &lt; len(list_foods) or i &lt; len(list_drinks):
                var_space = 1
                if i &lt;= 8:                        # To fix up to space indention in console or terminal by applying detection rule to figure out spacing for
                    var_space = 2

                if i &lt; len(list_foods):
                    food = " (" + str(i + 1) + ")" + " " * var_space + str(list_foods&#91;i]) + "  | " # Styling out the index number for the food or item and
                else:
                    food = " " * 36 + "| " # 36 is a constant for indention in console to fixup list in print
                if i &lt; len(list_drinks):
                    drink = "(" + str(41 + i) + ")" + " " + str(list_drinks&#91;i])
                else:
                    drink = ""
                print(food, drink)
                i += 1

            print("\n (M) MAIN MENU                   (P) PAYMENT                   (E) EXIT\n" + "_" * 72)

            input_1 = input("Please Select Your Operation: ").upper() #Handling Menu Selection
            if (input_1 == 'M'):
                print("\n" * 10)
                def_main() # Return to main menu by calling it out
                break
            if (input_1 == 'E'):
                print("*" * 32 + "THANK YOU" + "*" * 31 + "\n") # Handling Exit and print out thank you
                break
            if (input_1 == 'P'):
                print("\n" * 10)
                def_payment() # Handling payment || More details below
                break
            try:        #Cautions Error Handling to prevent program crashing and hand out exceptions as a readable error to notify user
                int(input_1)
                if ((int(input_1) &lt;= len(list_foods) and int(input_1) &gt; 0) or (int(input_1) &lt;= len(list_drinks) + 40 and int(input_1) &gt; 40)):
                     try:
                        print("\n" + "_" * 72 + "\n" + str(list_foods&#91;int(input_1) - 1])) # Handling Food Selection / The try/Execpt to handle out of inde
                     except:
                        pass
                     try:
                         print("\n" + "_" * 72 + "\n" + str(list_drinks&#91;int(input_1) - 41])) # Handling Drinks Selection / The try/Execpt to handle out of
                     except:
                        pass

                     input_2 = input("How Many You Want to Order?: ").upper() # Handling Quantity input
                     if int(input_2) &gt; 0:
                        list_item_order&#91;int(input_1) - 1] += int(input_2) # adding item to Orders Array
                        print("\n" * 10)
                        print("Successfully Ordered!")
                        def_food_drink_order() # Return food/drinks Menu
                        break
                     else:
                        print("\n" * 10 + "ERROR: Invalid Input (" + str(input_2) + "). Try again!")
            except:
                print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")

def def_other_services():
    while True:
        print("*" * 29 + "OTHER SERVICES" + "*" * 29)
        print(" |NO| |SERVICE NAME|      |PRICE|")  # Services Menu Structure

        i = 0
        while i &lt; len(list_services):
            print(" (" + str(81+ i) + ")" + " " + str(list_services&#91;i])) # Services starts from 81 + and now it is being enumarated into a list.

            i += 1

        print("\n (M) MAIN MENU                   (P) PAYMENT                   (E) EXIT\n" + "_" * 72)

        input_1 = input("Please Select Your Operation: ").upper()
        if (input_1 == 'M'):
            print("\n" * 10)
            def_main() # Navigate Back to main menu
            break
        if (input_1 == 'E'):
            print("*" * 32 + "THANK YOU" + "*" * 31 + "\n")
            break
        if (input_1 == 'P'):
            print("\n" * 10)
            def_payment() # navigate to payment
            break
        try:
            int(input_1)
            if (int(input_1) &gt; 80) and (int(input_1) &lt; 100):
                print("\n" * 10)
                print("Successfully Ordered: " + str(list_services&#91;int(input_1) - 81])) # Adding services to orders array (AND) encapsulate errors  with t
                list_item_order&#91;int(input_1) - 1] = 1
                def_other_services()
                break
            else:
                print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")
        except:
            print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")

def def_report():
    while True:
        print("*" * 33 + "REPORT" + "*" * 33 + "\n")
        file_report = open('files'+navigator_symbol+'report.fsd', 'r').read() # Reading out reports from report.fsd
        print(file_report)
        print("\n(M) MAIN MENU          (E) EXIT\n" + "_" * 72)
        input_1 = str(input("Please Select Your Operation: ")).upper()
        if (input_1 == 'M'):
            print("\n" * 10)
            def_main() # Navigate back to menu
            break
        elif (input_1 == 'E'):
            print("*" * 32 + "THANK YOU" + "*" * 31 + "\n") # Exit and break up the loop
            break
        else:
            print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")

def def_payment():
    while True:
        print("*" * 32 + "PAYMENT" + "*" * 33 + "\n") # Header &amp; Styling
        total_price = 0 # alloc/init a variable to handle total_price

        report_new = "\n\n\n" + " " * 17 + "*" * 35 + "\n" + " " * 17 + "DATE: " + str(datetime.datetime.now())&#91;:19] + "\n" + " " * 17 + "-" * 35 #buildin
        i = 0
        while i &lt; len(list_item_order): #Enumarating order array items and summing up its prices * quantities
            if(list_item_order&#91;i] != 0):
                if (i &gt;= 0) and (i &lt; 40):
                    report_new += "\n" + " " * 17 + str(list_foods&#91;i]) + "  x  " + str(list_item_order&#91;i]) # string appending the formated food name a
                    print(" " * 17 + str(list_foods&#91;i]) + "  x  " + str(list_item_order&#91;i])) #print it out
                    total_price += list_item_price&#91;i] * list_item_order&#91;i] # Calculating the total price for food
                if (i &gt;= 40) and (i &lt; 80):
                    report_new += "\n" + " " * 17 + str(list_drinks&#91;i - 40]) + "  x  " + str(list_item_order&#91;i])
                    print(" " * 17 + str(list_drinks&#91;i - 40]) + "   x  " + str(list_item_order&#91;i]))
                    total_price += list_item_price&#91;i] * list_item_order&#91;i] # Calculating the total price for drinks
                if (i &gt;= 80) and (i &lt; 100):
                    report_new += "\n" + " " * 17 + str(list_services&#91;i - 80])
                    print(" " * 17 + str(list_services&#91;i - 80]))
                    total_price += list_item_price&#91;i] * list_item_order&#91;i] # Calculating the total price for services
                i += 1
            else:
                i += 1
        ### Applying Discounts Ruless
        if total_price &gt; var_discount_3: ### price &gt; 5000
            total_price -= total_price * var_discount_3_rate # Discount fees from the total_price by 0.15 or 15%
            report_new += "\n" + " " * 17 + "-" * 35 + "\n" \
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_3_rate * 100) + "\n" \
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_3_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n" \
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)) + "\n" + " " * 17 + "*" * 35 # Round() to flour the float into an inter
            print(" " * 17 + "-" * 35 + "\n"
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_3_rate * 100) + "\n"
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_3_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n"
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)))
        elif total_price &gt; var_discount_2: ### price &gt; 3000
            total_price -= total_price * var_discount_2_rate # Discount fees from the total_price by 0.10 or 10%
            report_new += "\n" + " " * 17 + "-" * 35 + "\n" \
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_2_rate * 100) + "\n" \
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_2_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n" \
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)) + "\n" + " " * 17 + "*" * 35 # Round() to flour the float into an inter
            print(" " * 17 + "-" * 35 + "\n"
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_2_rate * 100) + "\n"
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_2_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n"
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)))
        elif total_price &gt; var_discount_1: ### price &gt; 200
            total_price -= total_price * var_discount_1_rate # Discount fees from the total_price by 0.05 or 5%
            report_new += "\n" + " " * 17 + "-" * 35 + "\n" \
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_1_rate * 100) + "\n" \
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_1_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n" \
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)) + "\n" + " " * 17 + "*" * 35 # Round() to flour the float into an inter
            print(" " * 17 + "-" * 35 + "\n"
                "" + " " * 17 + "DISCOUNT RATES:      % " + str(var_discount_1_rate * 100) + "\n"
                "" + " " * 17 + "DISCOUNT AMOUNTS:   RM " + str(round(total_price * var_discount_1_rate, 2)) + "\n" + " " * 17 + "_" * 35 + "\n"
                "" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)))
        else:
            report_new += "\n" + " " * 17 + "-" * 35 + "\n" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)) + "\n" + " " * 17 + "*" * 35
            print(" " * 17 + "_" * 35 + "\n" + " " * 17 + "TOTAL PRICES:       RM " + str(round(total_price, 2)))

        print("\n (P) PAY          (M) MAIN MENU          (R) REPORT          (E) EXIT\n" + "_" * 72)
        input_1 = str(input("Please Select Your Operation: ")).upper()
        if (input_1 == 'P'):
            print("\n" * 10)
            print("Successfully Paid!")
            file_report = open('files'+navigator_symbol+'report.fsd', 'a') # Save it into a file
            file_report.write(report_new)
            file_report.close()
            def_default() #Reset the program for the name order
        elif (input_1 == 'M'):
            print("\n" * 10)
            def_main() #Navigate back to the main menu
            break
        elif (input_1 == 'R'):
            print("\n" * 10)
            def_report() # Navigate to the reports
            break
        elif ('E' in input_1) or ('e' in input_1):
            print("*" * 32 + "THANK YOU" + "*" * 31 + "\n")
            break
        else:
            print("\n" * 10 + "ERROR: Invalid Input (" + str(input_1) + "). Try again!")
def_main() # Execute Main menu Loop
```

```
  File "<ipython-input-18-d842324d9f1f>", line 39
    break                                                     #Stop repeating Main Menu.
    ^
SyntaxError: 'break' outside loop
```

In [ ]: