|  |  |
| --- | --- |
| 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  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 #[food code](file:///C:\Users\Advaith\Downloads\food%20code.docx)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 of string length and align Prices to the most left of the terminal  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\_foods&#91;i]&#91;list\_foods&#91;i].index('RM'):])  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(list\_drinks&#91;i]&#91;list\_drinks&#91;i].index('RM'):])  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)) + str(list\_services&#91;i]&#91;list\_services&#91;i].index('RM'):])  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 array accordingly to a parrallel indexes  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 the string into an integer  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 which are the drinks  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 wich are Services  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 TWO DIGITS numbers  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 starting out from 1 for better human readability  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 index error as if it not exists in the array  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 index error as if it not exists in the array  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 try/except  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 #building up report string header  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 and formated order structure from quantity and final price  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 interger  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 interger  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 interger  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  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 of string length and align Prices to the most left of the terminal  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\_foods&#91;i]&#91;list\_foods&#91;i].index('RM'):])  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(list\_drinks&#91;i]&#91;list\_drinks&#91;i].index('RM'):])  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)) + str(list\_services&#91;i]&#91;list\_services&#91;i].index('RM'):])  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 array accordingly to a parrallel indexes  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 the string into an integer  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 which are the drinks  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 wich are Services  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 TWO DIGITS numbers  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 starting out from 1 for better human readability  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 index error as if it not exists in the array  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 index error as if it not exists in the array  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 try/except  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 #building up report string header  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 and formated order structure from quantity and final price  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 interger  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 interger  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 interger  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 LoopWelcome to Word  Instructions you can edit, share, and print | Microsoft Word logo |

Unlike old-school user guides, this doc is yours to tailor exactly for your needs. Reading it will teach you some basics about Word, but this document isn’t just for reading. It’s for editing too, so you can learn by doing.

For practice using Word features, watch for Try it text in red throughout this document.

|  |  |
| --- | --- |
| Learn Word with this interactive template | **Time saver:** If you’ve only got a minute and you want to see how this works, watch this Video: [Welcome to Word](https://www.microsoft.com/en-us/videoplayer/embed/RE1FGlM). |

# Write eloquently, with a little help

Word automatically checks spelling and grammar, and marks misspelled words with a red squiggly underline. Grammatical glitches get a blue double underline.

Try it: Put your cursor at the end of this paragraph, and hit Enter to start a new paragraph. Write a sentence with some spelling or grammatical mistakes, and press Enter to finish the paragraph.

Right-click the text that’s marked with underlines, or Press F7. Choose a suggestion to correct the mistakes.

Count on Word to count your words

Try it: Hit return after this line and type some words.

The status bar at the bottom of the window keeps a running count of the number of words in the document.

Example of word count displayed in the status bar

# Save this for later, access it anywhere

When you save this document in OneDrive, you’ll be able to open it anywhere: on your computer, tablet, or phone. Your changes will be saved automatically.



Try it: Select **File** > **Save As**, and then select OneDrive and give this document a name.

If you sign in to Office 365 on another device, this document will be in your list of recent files. You can pick up where you left off… even if you left the document open on the computer you’re using now.

Share and collaborate

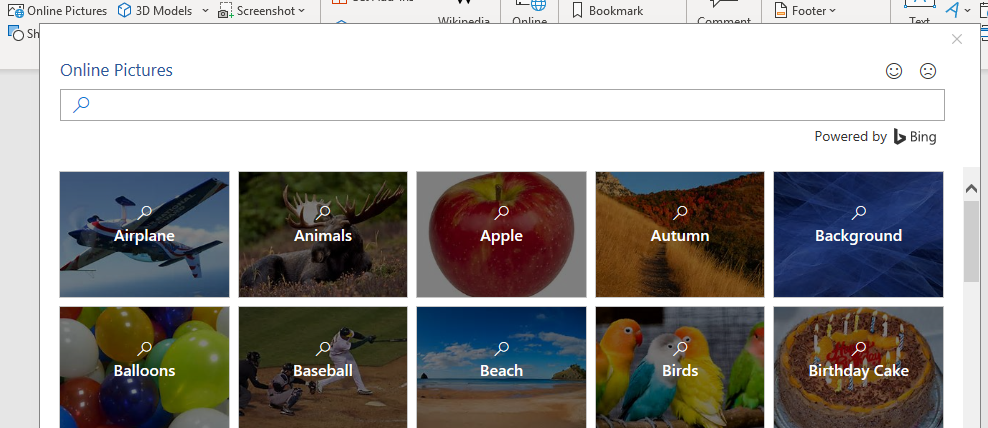
With this document saved in OneDrive, you can share it with others. They don’t even need Word to open it.

Try it: Select **Share**, and send a link to this document. (keyboard shortcut – Alt+F+Z or Alt+Z+S)

You can send the link by typing someone’s email address or by copying the link and pasting it into a message or chat. If you want them to read the document but not edit it, set their permission to view-only.

If they don’t have Word, the document will open in their web browser, in Word Online.

# Add visuals with pictures from the web



Word works with Bing to give you access to thousands of pictures you can use in your documents.

Try it: Hit enter after this line to make a blank line:

1. With your cursor in the blank space above, go to the Insert tab, select **Online Pictures**, and then search for something, like puppy clip art.
2. Select the picture you want, and select **Insert**.

# Make your meaning more visual by formatting text



To format text, select it, and then select a button in the **Font** or **Paragraph** area on the Home tab.

Try it: Select text in the lines below and choose formatting options so that the text is an example of the formatting it’s describing:

|  |  |
| --- | --- |
| Bold command icon | Bold (keyboard shortcut: Ctrl+B) |
| Italics command icon | Italic (keyboard shortcut: Ctrl+I) |
| Highlight command icon | Highlight |
| Font color command icon | Font color |
| Bullets command icon | Bullets |
| Numbering command icon | Numbering |

**Pro tip:** If you selected whole words for this exercise, did you notice that Word popped up a little toolbar, with the font formatting options?

|  |  |
| --- | --- |
| Contextual toolbar with commands for formatting text | Between that and keyboard shortcuts like Ctrl+B and Ctrl+I, you save time by not having to go up to the Home tab all the time. |

Make magic: use Heading styles

The heading for this part (“Make magic: use Heading styles”) looks the same as the other headings in this document, but it’s not as useful. It’s formatted with *font settings* (font, size, and color), while the other headings are formatted with a *Heading style* (Heading 1, to be exact).

|  |  |
| --- | --- |
| Heading showing expand/collapse button | See the little triangle when you mouse over those other headings?  You can collapse and expand everything under a heading, like an outline. But this one’s not working. Let’s fix it. |

Try it: Apply the **Heading 1** style:

1. Put your cursor somewhere in the heading above (“Make magic: use Heading styles”) – don’t select anything.
2. On the **Home** tab, find **Styles**, and select **Heading 1** (keyboard shortcut Ctrl+Alt+1).

Ta-da! Now it looks like a heading, and acts like one too.

Give your doc an instant makeover



Style sets and themes let you completely change the look of your document in an instant. They work best when your document is formatted with styles (so it’s good that we fixed that Heading style, above).

Try it: Explore style sets and themes:

1. On the **Design** tab, select **Themes**, and choose a theme from the drop-down.  
   Notice that the gallery of style sets updates to reflect the theme you picked.
2. Select any theme you like from the drop-down and click to apply.

Just for fun

If you have a little more time, give these a try.

Try it: Find and replace

Press **Ctrl+H** and use Find/Replace to replace all the instances of Try it with I tried it.



Try it: Wrap text around pictures

|  |  |
| --- | --- |
| Some of the pictures in this document are placed beside a paragraph of text. Try exploring text wrapping: Select the picture here, go to **Format** > **Wrap Text**, and use the up/down arrow keys to move between the options for a preview of the results. | Example of text wrapping around a picture |

Get help with Word



The **Tell me** search box takes you straight to commands and Help in Word.

Try it: Get help:

1. Go to **Tell me what you want to do** at the top of the window.
2. Type what you want to do.

For example, type:

* **Add watermark** to quickly get to the watermark command.
* **Help** to go to Word help.
* **Training** to see the list of Word training courses.
* **What’s new** for a list of the most recent updates to Word

[](https://go.microsoft.com/fwlink/?linkid=854192)