Προαιρετική Εργασία σε Python 2019 - 2020

Θωμάς Γεώργιος, 1059634

2020

Περίληψη

Στην παρούσα εργασία με τη χρήση της Python πραγματοποιήθηκε κατέβασμα και αποθήκευση αρχείων από το διαδίκτυο. Ύστερα, έγινε εξαγωγή των απαραίτητων στοιχείων από τα αρχεία αυτά σε μια βάση δεδομένων. Από εκεί, έγινε επεξεργασία και υπολογίστηκαν οι τελικές τιμές των ζητουμένων. Από τις τιμές αυτές πραγματοποιήθηκε εξαγωγή διαγραμμάτων αλλά και των csv αρχείων.

Εισαγωγή

Η εργασία αυτή πραγματοποιήθηκε σε λειτουργικό Linux και συγκεκριμένα σε Ubuntu 18.04. Έγινε χρήση της Python 3.6.9. Ως text editor χρησιμοποιήθηκε το Visual Studio Code και ως βάση δεδομένων η SQLite. Επιπλέον, για την πραγματοποίηση της εργασίας χρησιμοποιήθηκαν και οι κατάλληλες βιβλιοθήκες, οι οποίες φαίνονται στην αρχή του κάθε αρχείου. Τέλος, μέσα στον κώδικα υπάρχουν σχόλια καθώς και συναρτήσεις εκτύπωσης για την καλύτερη κατανόηση του προγράμματος.

1 Κατέβασμα, αποθήκευση αρχείων και εξαγωγή δεδομένων

Στον παρακάτω κώδικα υλοποιήθηκαν δύο συναρτήσεις. Η πρώτη κατεβάζει και αποθηκεύει τα αρχεία στον υπολογιστή. Με τη χρήση βιβλιοθηκών γίνεται σύνδεση στην ιστοσελίδα. Από εκεί αναζητείται το download link με βάση τον τίτλο του, γίνεται κατέβασμα και ύστερα αποθήκευση. Οι πληροφορίες που χρειάζονται για να απαντηθούν τα ερωτήματα βρίσκονται όλες στα τέσσερα αρχεία που κατεβάζονται. Η δεύτερη συνάρτηση που υλοποιείται είναι αυτή για την εξαγωγή των απαραίτητων τιμών σε μία βάση δεδομένων αφού πρώτα φιλτραριστούν και κρατηθούν μόνο αυτά που χρειάζονται.

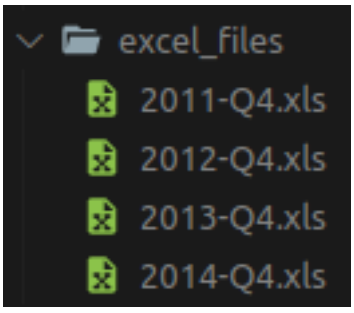
```
1  from bs4 import BeautifulSoup # Scrap urls from webpages
2  import urllib.request # Make url requests
3  import requests # Open and store files from urls
4  import calendar # Get the months
5  import sqlite3 # Manage a database
6  import xlrd # Read excel files
7  import os # Manage directories
8  import re # Regeular expressions
9
10
```

```
# Download and store excel files
def download():
    # The main url to get download links
    main_url = "https://www.statistics.gr/en/statistics/-/publication/STO04/201X-Q4"

    for year in range(1, 5):
        new_url = main_url.replace("X", str(year)) # Change the year

        file_name = new_url.split("/")[-1] + ".xls" # Get the file name
        print("Downloading " + file_name + "...")

        # Open the corresponding page
        html_page = urllib.request.urlopen(new_url)

        # Parse the page
        soup = BeautifulSoup(html_page, "html.parser")

        # Find the first "a" tag with the given text
        link = soup.find("a", text="Arrivals of non-residents from abroad, by country of residence and by means of transport ")

        # Get the download link from the page
        url = link.get("href")

        # Get the relative path filename in respect to cwd
        rel_path_filename = "../" + "excel_files" + "/" + file_name

        # Open the link, download the file and store it in the given folder
        r = requests.get(url, allow_redirects=True)
        open(rel_path_filename, "wb+").write(r.content)

    print("All files downloaded.")


# Export the necessary information from excel files to database
def export_to_db():
    print("Exporting to database...")

    # Create/connect to database
    conn = sqlite3.connect("../tourism.db")

    # All excel files
    excel_files = os.listdir("../excel_files/")

    c = conn.cursor()

    # Create table
    c.execute('''CREATE TABLE IF NOT EXISTS statistics
                 (Country TEXT, Air REAL, Railway REAL, Sea REAL, Road REAL, Month TEXT, Year INTEGER, unique(Country, Month, Year))''')

    # For every excel file
    for excel_file in excel_files:

        path = "../excel_files/" + excel_file

        # Open the file and get the sheets
```

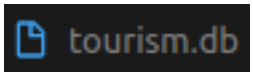
```
66          book = xlrd.open_workbook(path)
67
68          # For every sheet within excel file
69          for i, sheet in enumerate(book.sheets()):
70              shown = 0
71              # Get the first table of the sheet and add it to
    database
72              for row_num in range(sheet.nrows):
73                      row_value = sheet.row_values(row_num)
74                      if row_value[0] == " - EUROPEAN UNION":
75                          shown += 1
76                      r = re.compile(r"\d.")
77                      if shown == 1 and r.match(row_value[0]):
78                          c.execute("INSERT INTO statistics VALUES
    (?, ?, ?, ?, ?, ?, ?)",
79                                    (row_value[1], row_value[2],
    row_value[3], row_value[4], row_value[5], calendar.month_name[i
    +1], excel_file.split("-")[0]))
80                      if shown == 2:
81                          break
82
83      # Fix empty values
84      means_of_transport = ["Air", "Railway", "Sea", "Road"]
85      for mean in means_of_transport:
86          c.execute("UPDATE statistics SET {} = 0 WHERE {} = ''".
    format(mean, mean))
87
88
89      conn.commit() # Commit changes
90
91      conn.close() # Close connection
92
93      print("Done.")
```

Listing 1: files.py

Τα αποτελέσματα ύστερα από την εκτέλεση των συναρτήσεων αυτών είναι η αποθήκευση των αρχείων στον υπολογιστή και η δημιουργία της βάσης δεδομένων από τα αρχεία αυτά, όπως φαίνονται στα παρακάτω screenshots.



(α′) Αρχεία Excel



(β′) Βάση Δεδομένων

Σχήμα 1: Αρχεία που δημιουργήθηκαν

2 Επεξεργασία δεδομένων και εξαγωγή αποτελεσμάτων σε αρχεία csv

Στον παρακάτω κώδικα υλοποιήθηκαν πέντε συναρτήσεις. Στις τέσσερις από αυτές φορτώνονται τα δεδομένα από την βάση και επεξεργάζονται ώστε σαν αποτελέσματα να περιέχουν την πληροφορία που αφορά το κάθε ερώτημα. Η επεξεργασία των δεδομένων έγινε με την χρήση της βιβλιοθήκης pandas. Η πέμπτη συνάρτηση εξάγει και αποθηκεύει τα τελικά δεδομένα σε μορφή csv.

```
import pandas as pd # Analyze data
import calendar # Get the months
import sqlite3 # Manage the datatbase


def total_arrivals():
    print("Calculating Total Arrivals...")

    # Connect to database
    conn = sqlite3.connect("../tourism.db")

    # Get the dataframe from the database
    df = pd.read_sql_query("SELECT * FROM statistics", conn)

    # Get the total arrivals for each year
    total_df = pd.DataFrame({"Total Arrivals" : df.groupby(by = "
    Year")[["Air", "Railway", "Sea", "Road"]].sum().sum(axis=1)}).
    reset_index()

    # Close the connection
    conn.close()

    print("Done.")

    return total_df


def total_arrivals_by_country():
    print("Calculating Total Arrivals by Country...")

    # Connect to database
    conn = sqlite3.connect("../tourism.db")

    # Get the dataframe from the database
    df = pd.read_sql_query("SELECT * FROM statistics", conn)

    # Get the total arrivals for each year
    total_df = pd.DataFrame({"Total Arrivals" : df.groupby(by = "
    Country")[["Air", "Railway", "Sea", "Road"]].sum().sum(axis=1)
    }).reset_index()

    # Sort values by total arrivals
    total_df = total_df.sort_values(by = ["Total Arrivals"],
    ascending = False).head(4)

    # Close the connection
    conn.close()

    print("Done.")

    return total_df

```

```
48
49 def total_arrivals_by_means_of_transport():
50     print("Calculating Total Arrivals by Means of Transport...")
51
52     # Connect to database
53     conn = sqlite3.connect("../tourism.db")
54
55     # Get the dataframe from the database
56     df = pd.read_sql_query("SELECT * FROM statistics", conn)
57
58     # Get the total arrivals for each year
59     total_df = pd.DataFrame({"Total Arrivals" : df[["Air", "Railway
       ", "Sea", "Road"]].sum()}).reset_index()
60
61     # Rename the column
62     total_df = total_df.rename(columns = {"index":"Means of
       Transport"})
63
64     # Sort values by total arrivals
65     total_df = total_df.sort_values(by = ["Total Arrivals"],
       ascending = False)
66
67     # Close the connection
68     conn.close()
69
70     print("Done.")
71
72     return total_df
73
74
75 def total_arrivals_by_quarter():
76     print("Calculating Total Arrivals by Quarter...")
77
78     # Connect to database
79     conn = sqlite3.connect("../tourism.db")
80
81     # Get the dataframe from the database
82     df = pd.read_sql_query("SELECT * FROM statistics", conn)
83
84     # Add quarter column
85     df["Quarter"] = ""
86
87     # Make the dataframe
88     total_df = pd.DataFrame(df)
89
90     # Update the quarter column
91     for month in calendar.month_name:
92         if month in calendar.month_name[1:4]:
93             total_df.loc[total_df["Month"] == month, "Quarter" ] =
       "Q1"
94         elif month in calendar.month_name[4:7]:
95             total_df.loc[total_df["Month"] == month, "Quarter" ] =
       "Q2"
96         elif month in calendar.month_name[7:10]:
97             total_df.loc[total_df["Month"] == month, "Quarter" ] =
       "Q3"
98         elif month in calendar.month_name[10:13]:
99             total_df.loc[total_df["Month"] == month, "Quarter" ] =
       "Q4"
100
101    # Get the total arrivals for each year's quarter
```

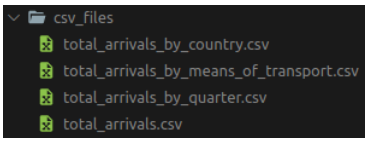
```
    total_df = pd.DataFrame({"Total Arrivals" : df.groupby(by = ["
    Year", "Quarter"])[["Air", "Railway", "Sea", "Road"]].sum().sum
    (axis=1)}).reset_index()

    print("Done.")

    return total_df


# Export csv file from pandas' dataframe
def export_to_csv(df, csv_name):
    print("Exporting " + csv_name + "...")
    csv_path = "../csv_files/" + csv_name # Path to store the csv
    file
    df.to_csv(csv_path, index = False) # Save the csv file
    print("Done.")
```

Listing 2: dataframe.py

Τα αποτελέσματα ύστερα από την εκτέλεση των συναρτήσεων αυτών, είναι η επιστροφή των τελικών αποτελεσμάτων σε μορφή dataframe καθώς και η εξαγωγή τους σε αρχεία csv, όπως φαίνονται στα παρακάτω screenshots.

```
Total Arrivals
   Year  Total Arrivals
0  2011    1.642725e+07
1  2012    1.551762e+07
2  2013    1.792327e+07
3  2014    2.203346e+07

Total Arrivals by Country
                    Country  Total Arrivals
17                  Germany    9.076042e+06
36  Other European countries    7.906535e+06
54           United Kingdom    7.614748e+06
16                   France    4.742140e+06

Total Arrivals by Means of Transport
  Means of Transport  Total Arrivals
0                Air    4.902487e+07
3               Road    1.961691e+07
2                Sea    3.249157e+06
1            Railway    1.065926e+04

Total Arrivals by Quarter
    Year Quarter  Total Arrivals
0   2011      Q1    1.108387e+06
1   2011      Q2    4.195768e+06
2   2011      Q3    8.925699e+06
3   2011      Q4    2.197393e+06
4   2012      Q1    9.785586e+05
5   2012      Q2    3.849245e+06
6   2012      Q3    8.655186e+06
7   2012      Q4    2.034632e+06
8   2013      Q1    1.023354e+06
9   2013      Q2    4.397478e+06
10  2013      Q3    1.011676e+07
11  2013      Q4    2.385673e+06
12  2014      Q1    1.186900e+06
13  2014      Q2    5.077136e+06
14  2014      Q3    1.272292e+07
15  2014      Q4    3.046501e+06
```

(α′) Εκτύπωση dataframes



(β′) Αρχεία csv

Σχήμα 2: Αποτελέσματα των συναρτήσεων

3 Εξαγωγή διαγραμμάτων

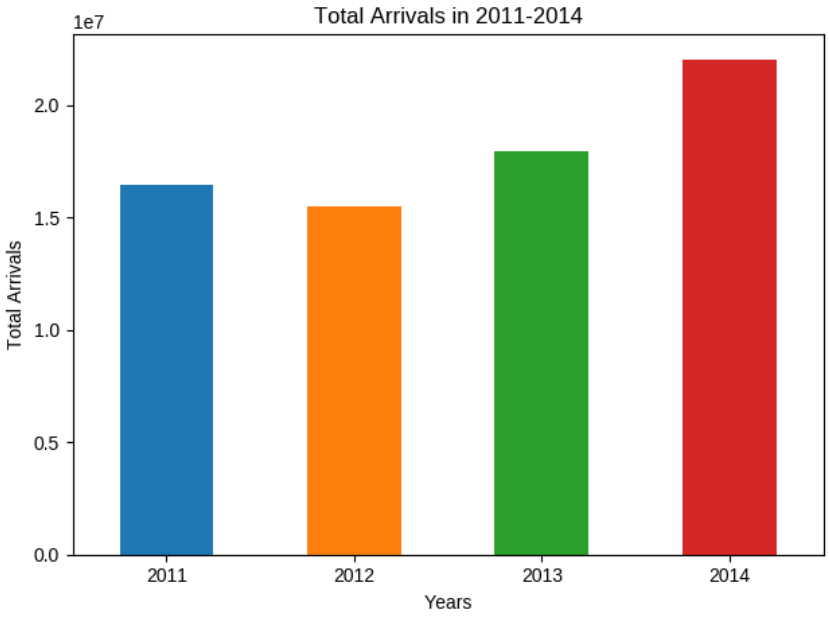
Στον παρακάτω κώδικα υλοποιήθηκαν τέσσερις συναρτήσεις. Η κάθε μία εξάγει γράφημα από το αντίστοιχο dataframe του κάθε ερωτήματος. Με τη χρήση των βιβλιοθηκών pandas και matplotlib γίνεται κατάλληλο styling για την καλύτερη κατανόηση των γραφημάτων και ύστερα αποθήκευση στον υπολογιστή.

```
import matplotlib.pyplot as plt # Create diagrams
import pandas as pd # Analyze data
import textwrap as tw # Text wrapping

# Diagram for total arrivals
def export_total_arrivals(df, diagram_name):
    print("Exporting " + diagram_name + "...")
    rel_path_filename = "../diagrams/" + diagram_name
    df.plot(kind = "bar", x = "Year", y = "Total Arrivals", title =
     "Total Arrivals in 2011-2014", legend = False) # Plotting the
    dataframe
    plt.style.use("seaborn-dark") # Color style
    plt.ylabel("Total Arrivals") # Label for y axis
    plt.xlabel("Years") # Label for x axis
    plt.xticks(rotation = 0) # Rotation for x axis' labels
    plt.tight_layout() # Fit labels
    plt.savefig(rel_path_filename) # Save the diagram
    print("Done.")


# Diagram for total arrivals by country
def export_total_arrivals_by_country(df, diagram_name):
    print("Exporting " + diagram_name + "...")
    rel_path_filename = "../diagrams/" + diagram_name
    df.plot(kind = "bar", x = "Country", y = "Total Arrivals",
    title = "Total Arrivals by Country in 2011-2014", legend =
    False) # Plotting the dataframe
    plt.style.use("seaborn-dark") # Color style
    plt.ylabel("Total Arrivals") # Label for y axis
    plt.xlabel("Countries") # Label for x axis

    text = '''\
    Note: Other European Countries are European countries besides
    Austria, Belgium, Bulgaria, Denmark, Estonia, Ireland,
    Spain, Italy, Croatia, Cyprus, Latvia, Lithuania, Luxembourg,
    Malta, Netherlands, Hungary, Poland, Portugal,
    Romania, Slovakia, Slovenia, Sweden, Czech Republic, Finland,
    Albania, Switzerland, Norway, Iceland, Russia, Servia.
    '''

    # Format the text
    fig_txt = tw.fill(tw.dedent(text.rstrip()), width=80)

    # Place the text
    plt.figtext(0.5, -0.12, fig_txt, horizontalalignment = "center"
    ,
                fontsize = 8, multialignment = "left",
                bbox=dict(boxstyle = "round", facecolor = "#D8D8D8"
    ,
                          ec = "0.5", pad = 0.5, alpha = 1),
    fontweight = "bold")

    plt.xticks(rotation = 45) # Rotation for x axis' labels
    plt.tight_layout() # Fit labels
```

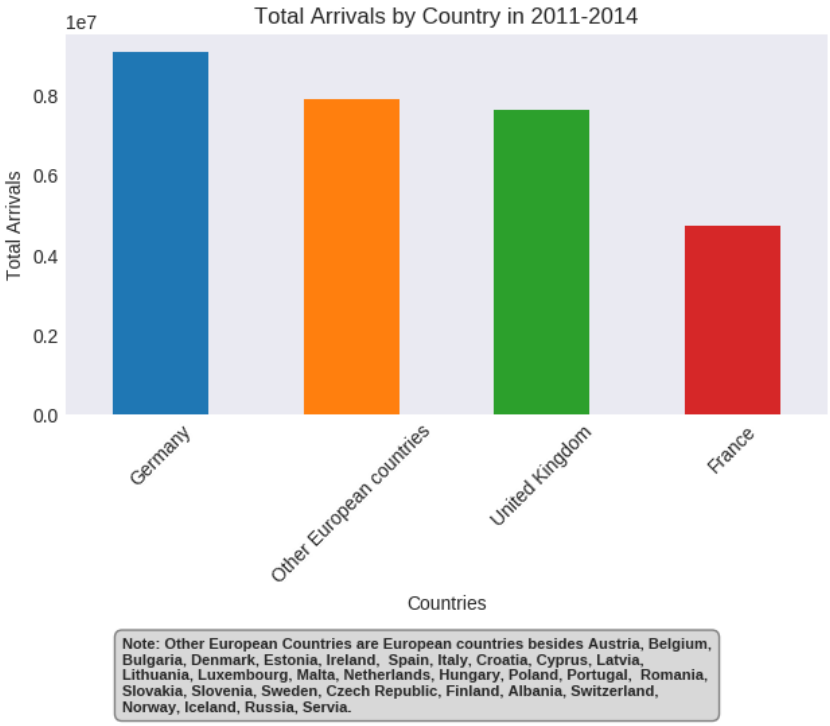
```
45      plt.savefig(rel_path_filename, bbox_inches = "tight") # Save
        the diagram
46      print("Done.")
47
48
49
50  # Diagram for total arrivals by means of transport
51  def export_total_arrivals_by_means_of_transport(df, diagram_name):
52      print("Exporting " + diagram_name + "...")
53      rel_path_filename = "../diagrams/" + diagram_name
54      df.plot(kind = "bar", x = "Means of Transport", y = "Total
        Arrivals", title = "Total Arrivals by Means of Transport in
        2011-2014", legend = False) # Plotting the dataframe
55      plt.style.use("seaborn-dark") # Color style
56      plt.ylabel("Total Arrivals") # Label for y axis
57      plt.xlabel("Means of Transport") # Label for x axis
58      plt.xticks(rotation = 0) # Rotation for x axis' labels
59      plt.tight_layout() # Fit labels
60      plt.savefig(rel_path_filename) # Save the diagram
61      print("Done.")
62
63
64  # Diagram for total arrivals by quarter
65  def export_total_arrivals_quarter(df, diagram_name):
66      print("Exporting " + diagram_name + "...")
67      rel_path_filename = "../diagrams/" + diagram_name
68      df.pivot("Year", "Quarter", "Total Arrivals").plot(kind="bar",
        title = "Total Arrivals by Quarter in 2011-2014") # Plotting
        the dataframe
69      plt.style.use("seaborn-dark") # Color style
70      plt.ylabel("Total Arrivals") # Label for y axis
71      plt.xlabel("Years") # Label for x axis
72      plt.xticks(rotation = 0) # Rotation for x axis' labels
73      plt.tight_layout() # Fit labels
74      plt.savefig(rel_path_filename) # Save the diagram
75      print("Done.")
```

Listing 3: diagram.py

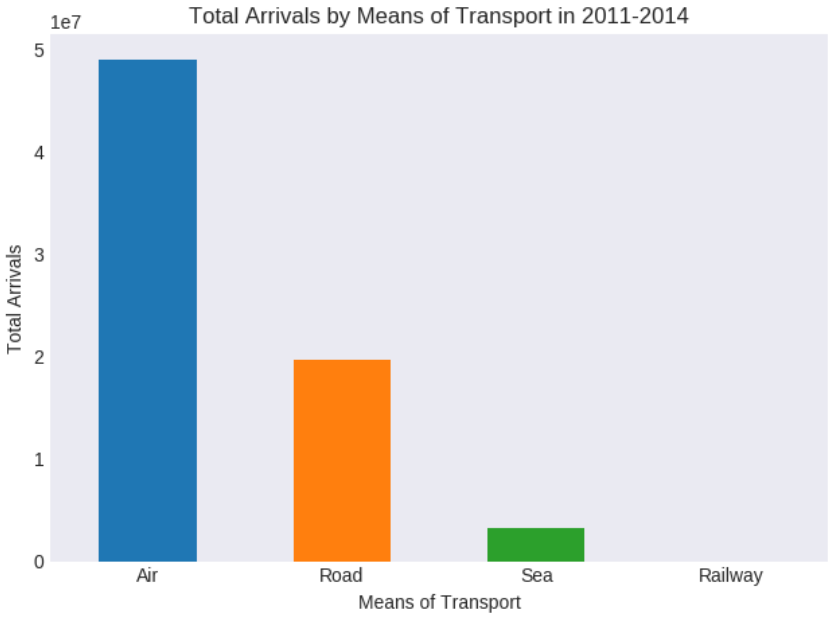
Το αποτέλεσμα ύστερα από την εκτέλεση των συναρτήσεων αυτών είναι η εξαγωγή των τεσσάρων διαγραμμάτων που απαντούν στα αντίστοιχα ερωτήματα.



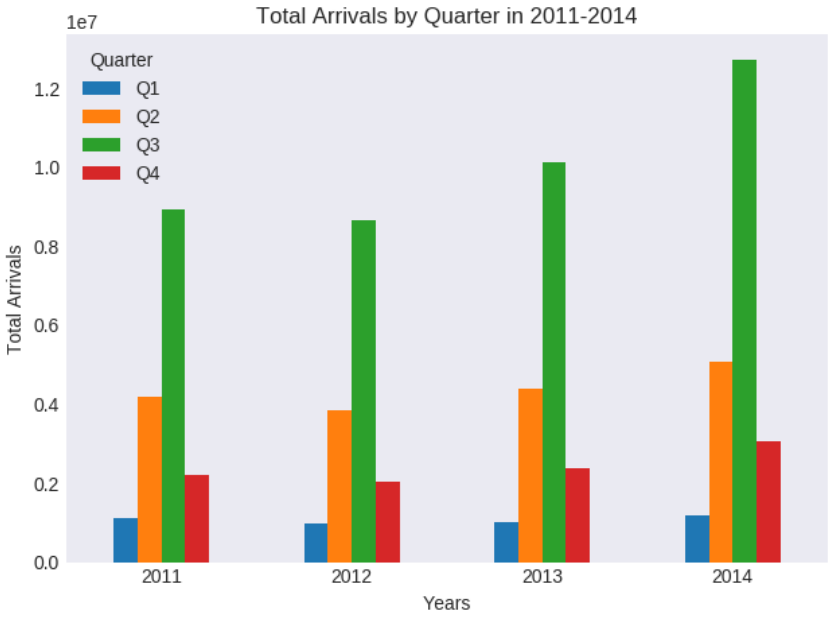
Σχήμα 3: Συνολικές αφίξεις



Σχήμα 4: Συνολικές αφίξεις ανά χώρα



Σχήμα 5: Συνολικές αφίξεις ανά μέσο μεταφοράς



Σχήμα 6: Συνολικές αφίξεις ανά τρίμηνο

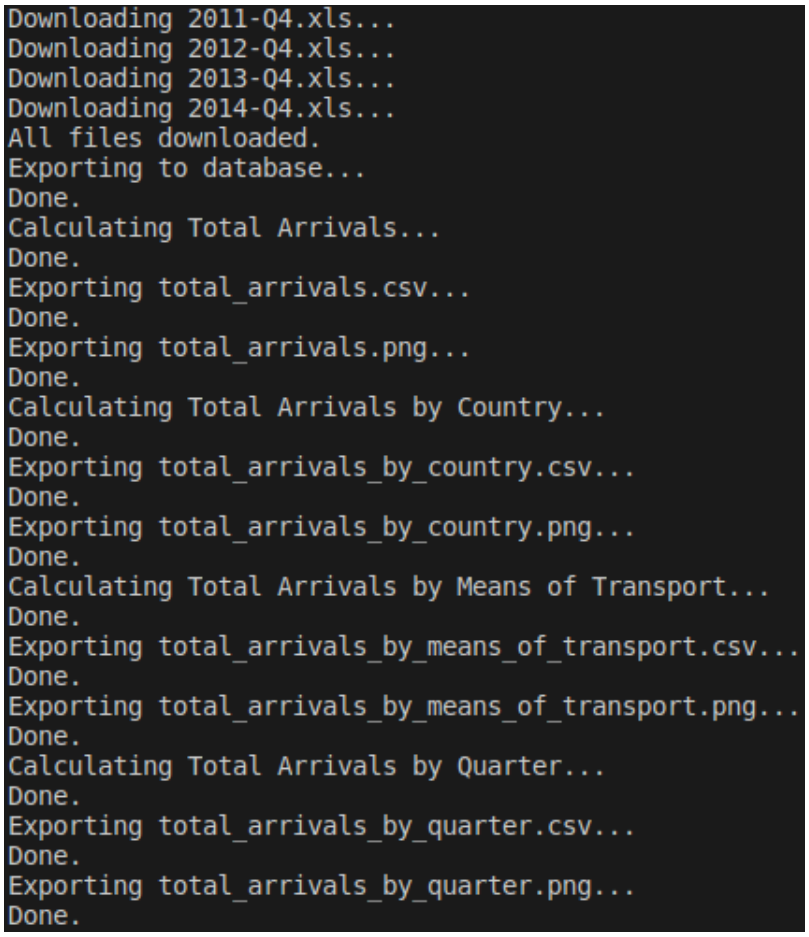
4 Χρήση των συναρτήσεων για την εξαγωγή των αποτελεσμάτων

Στον παρακάτω κώδικα καλούνται όλες οι συναρτήσεις με τα σωστά ορίσματα και τη σωστή σειρά έτσι ώστε να εξαχθούν τα ζητούμενα αποτελέσματα. Το αρχείο αυτό είναι το κύριο και είναι αυτό που χρειάζεται να εκτελεστεί για να παρθούν τα αποτελέσματα. Το αρχείο αυτό χρειάζεται να εκτελεστεί μία φορά. Αν εκτελεστεί πάνω από μία φορά θα βγάλει σφάλμα καθώς θα προσπαθήσει να εισάγει στη βάση ίδιες τιμές.

```
import dataframe
import diagram
import files
import os

# Get this file directory and set it as cwd
dir_path = os.path.dirname(os.path.realpath(__file__))
os.chdir(dir_path)

files.download()

files.export_to_db()

# Total arrivals for each year
total_arrivals = dataframe.total_arrivals()
dataframe.export_to_csv(total_arrivals, "total_arrivals.csv")
diagram.export_total_arrivals(total_arrivals, "total_arrivals.png")

# Total arrivals by country
total_arrivals_by_country = dataframe.total_arrivals_by_country()
dataframe.export_to_csv(total_arrivals_by_country, "
    total_arrivals_by_country.csv")
diagram.export_total_arrivals_by_country(total_arrivals_by_country,
     "total_arrivals_by_country.png")

# Total arrivals by means of transport
total_arrivals_by_means_of_transport = dataframe.
    total_arrivals_by_means_of_transport()
dataframe.export_to_csv(total_arrivals_by_means_of_transport, "
    total_arrivals_by_means_of_transport.csv")
diagram.export_total_arrivals_by_means_of_transport(
    total_arrivals_by_means_of_transport, "
    total_arrivals_by_means_of_transport.png")

# Total arrivals by each year's quarter
total_arrivals_by_quarter = dataframe.total_arrivals_by_quarter()
dataframe.export_to_csv(total_arrivals_by_quarter, "
    total_arrivals_by_quarter.csv")
diagram.export_total_arrivals_quarter(total_arrivals_by_quarter, "
    total_arrivals_by_quarter.png")
```

Listing 4: main.py

Ύστερα από την εκτέλεση του κώδικα αυτού, εξάγονται τα αρχεία csv και τα διαγράμματα, τα οποία έχουν παρουσιαστεί προηγουμένως. Στο παρακάτω screenshot φαίνονται τα εκτυπωμένα μηνύματα του terminal.



Σχήμα 7: Εκτύπωση Μηνυμάτων