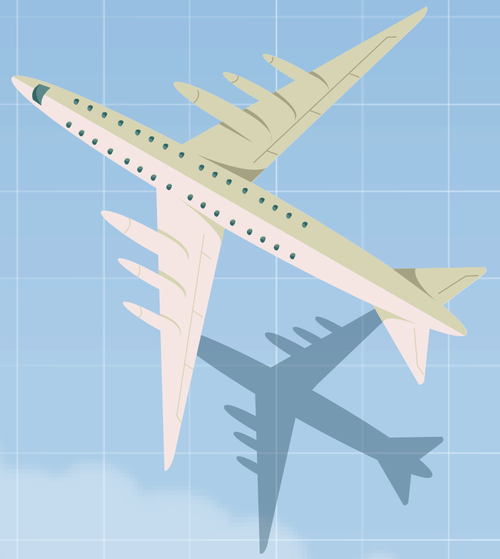
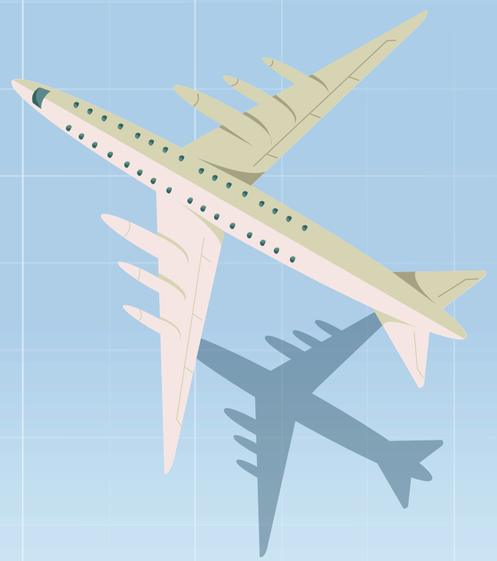
FLYING FREE:

DATA ANALYSIS WITH PYTHON FOR AERONAUTICAL PLANNING



JOAQUÍN (QUINO) - AIR HORIZONT

WHAT IS A FLIGHT DISPATCHER?

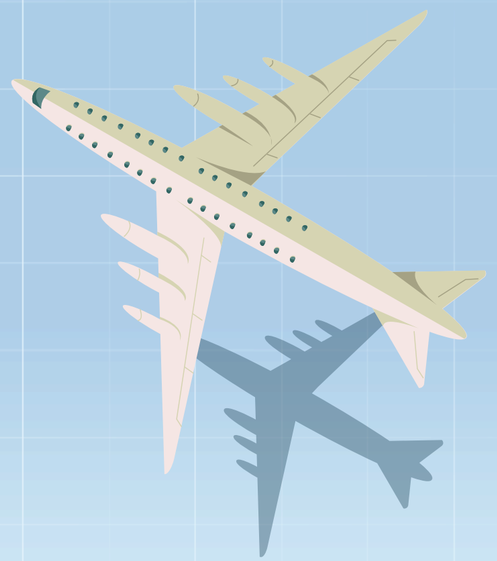


AIR TRAFFIC CONTROLLER

≠

FLIGHT DISPATCHER

WHAT IS A FLIGHT DISPATCHER?

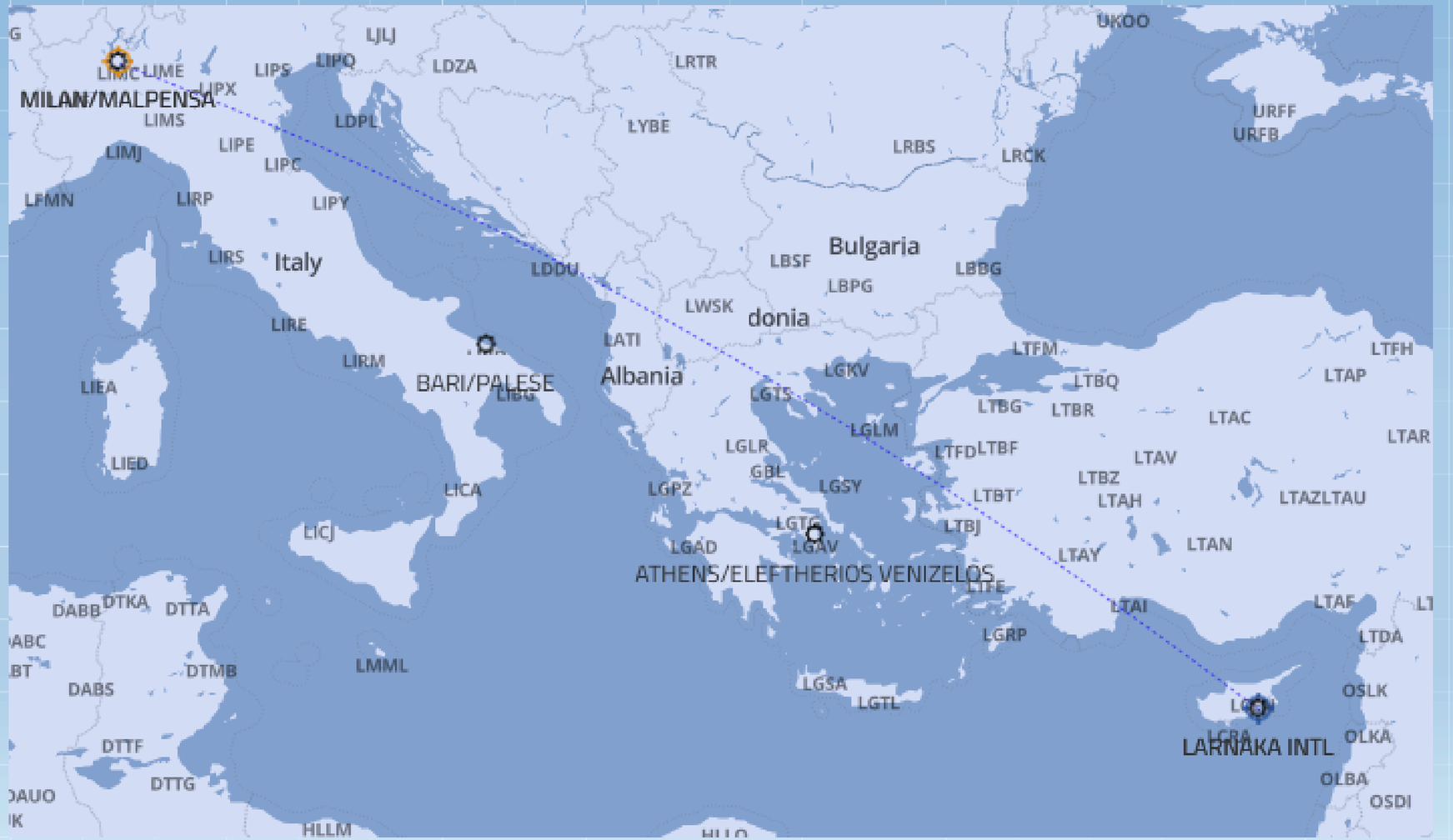


FLIGHT PLANNER

=

FLIGHT DISPATCHER

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

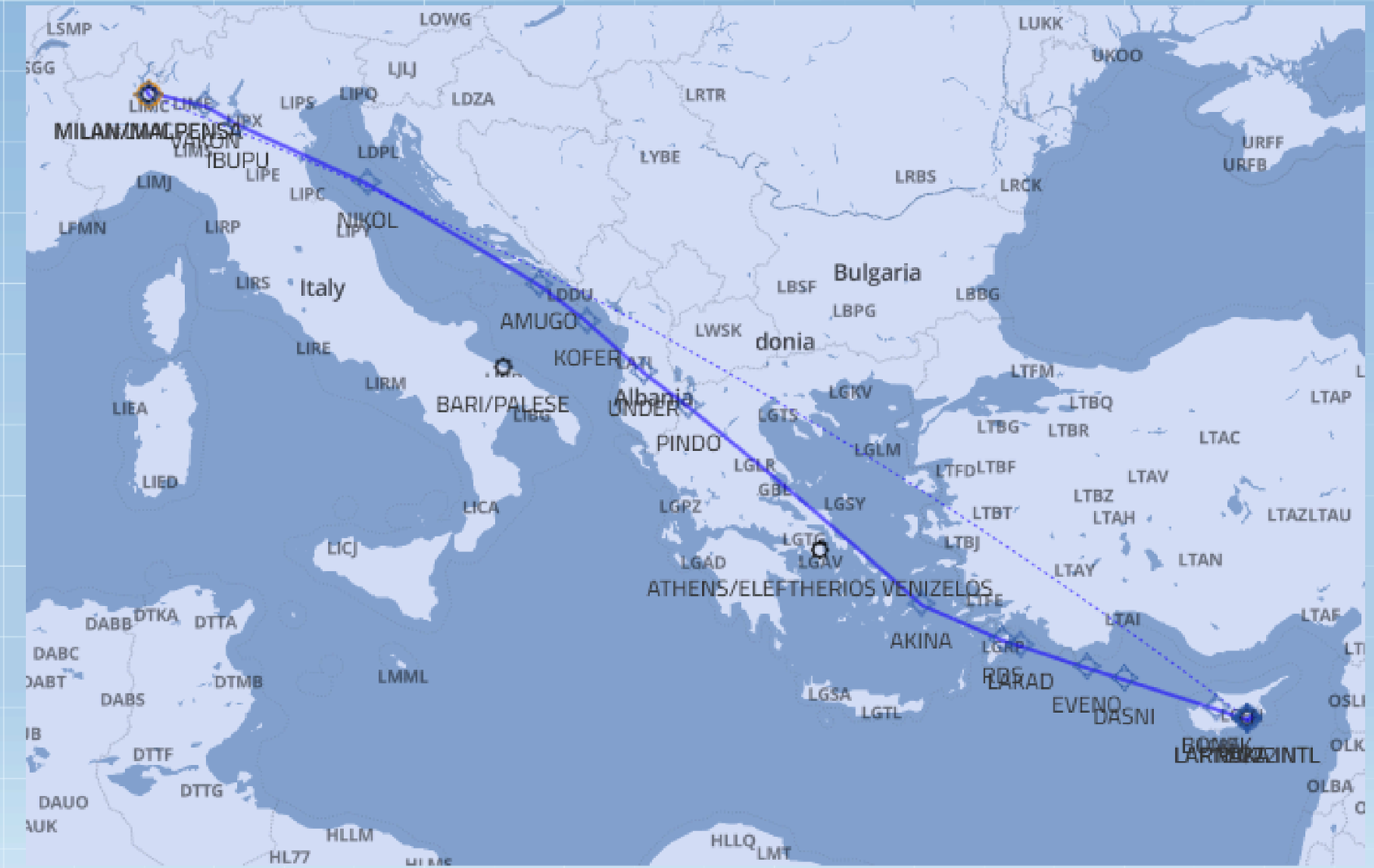
-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

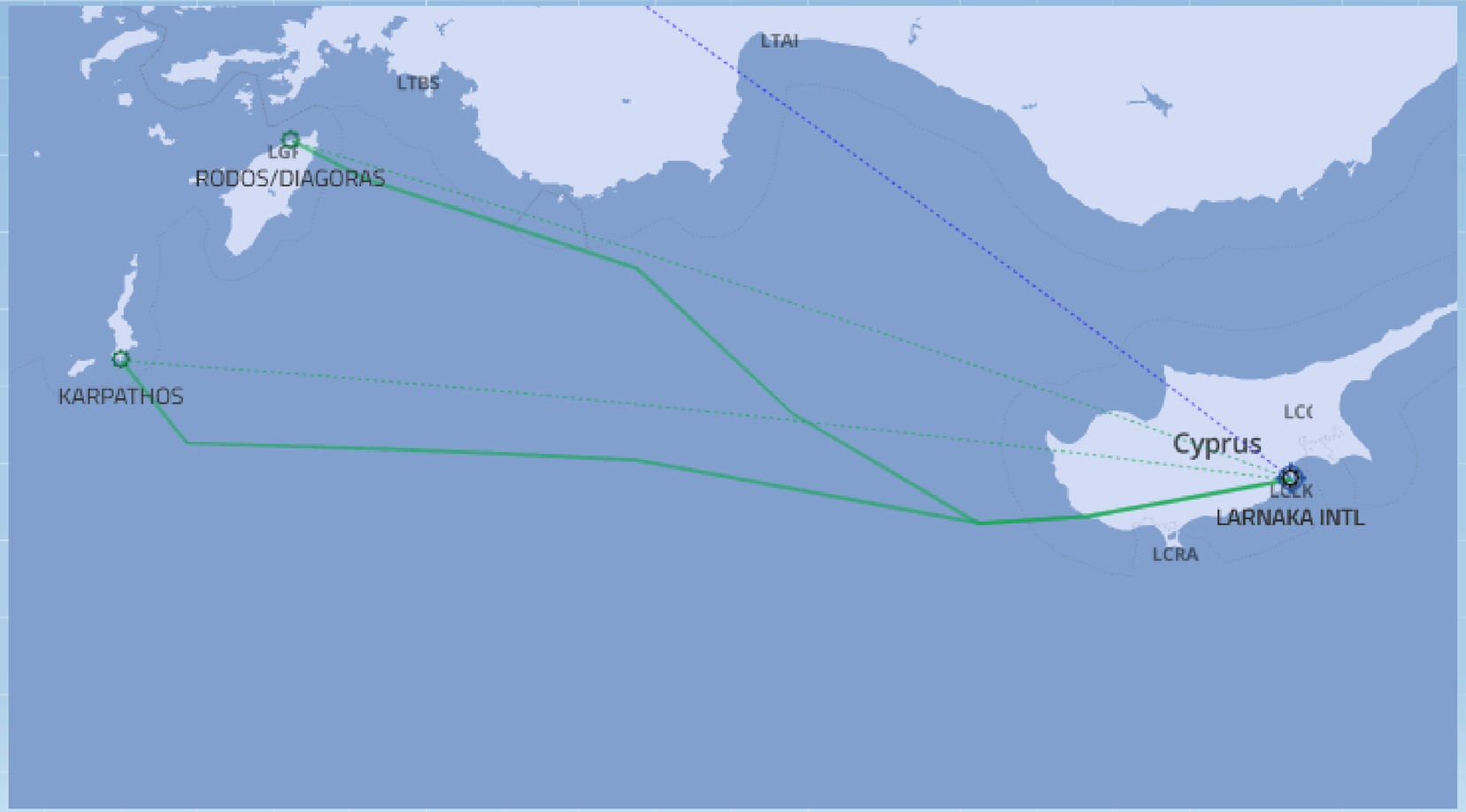
-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

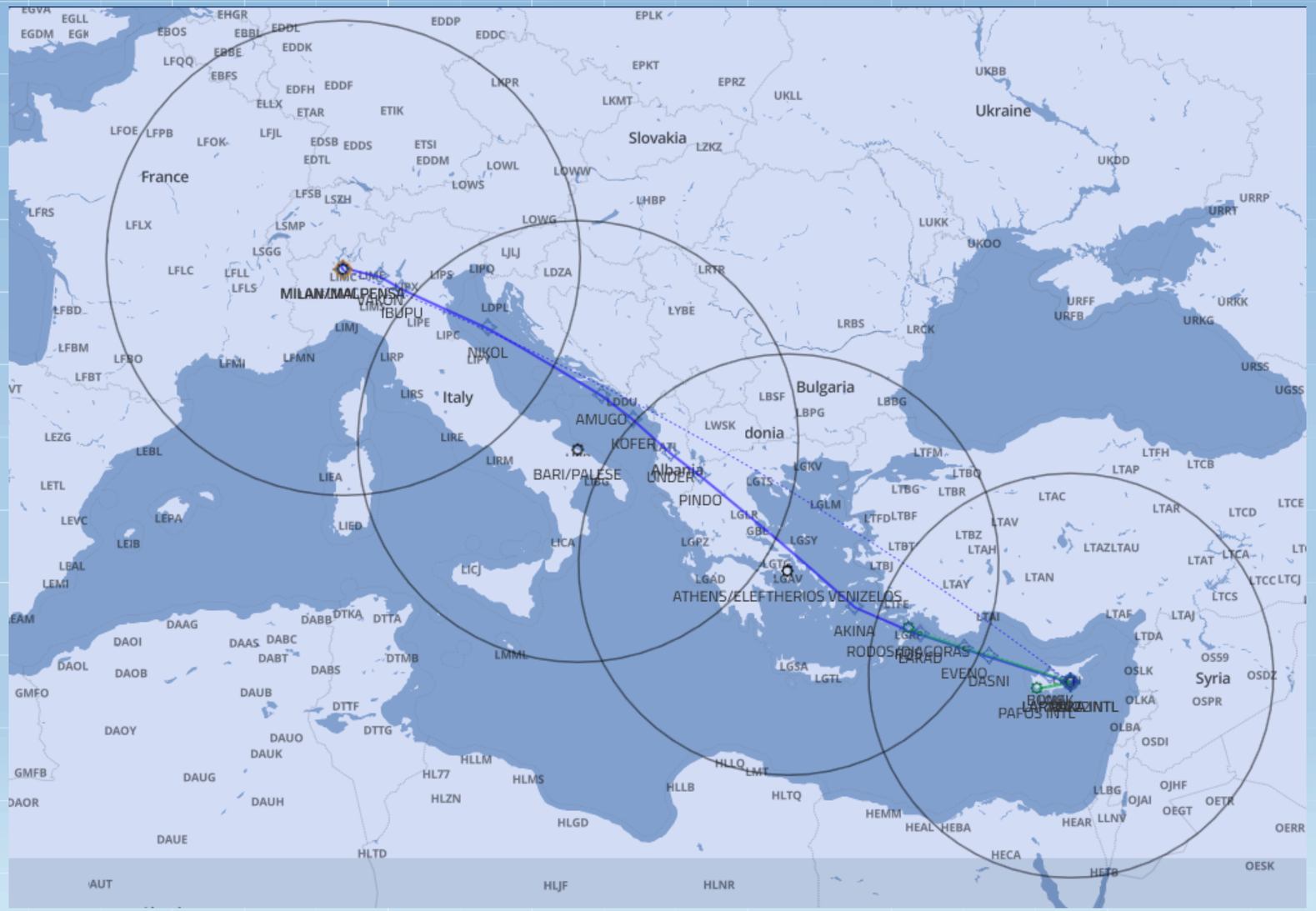
-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:

(WX search performed 2025-07-04 03:48:15 UTC.
Searched for METAR and TAF.)

Airport LIMC - MXP - MILAN/MALPENSA RWY 17L 17R 35L 35R

```
METAR  040320Z  35007KT CAVOK 21/19 Q1019 NOSIG=
TAF    032300Z  0400/0506 VRB05KT 9999 FEW050 BECMG 0411/0413 22010KT BECMG
       0416/0418 VRB05KT TEMPO 0504/0506 TSRA=
```

Airport LCLK - LCA - LARNAKA INTL RWY 04 22

```
METAR  040330Z  32005KT 280V360 CAVOK 27/11 Q1007 NOSIG=
TAF    032330Z  0400/0500 32009KT 9999 FEW020 BECMG 0404/0407 04014KT BECMG
       0411/0414 22012KT BECMG 0418/0421 33006KT=
```

Airport LIBD - BRI - BARI/PALESE RWY 07 25

```
METAR  040320Z  21005KT CAVOK 22/09 Q1017=
TAF    032300Z  0400/0424 VRB05KT CAVOK BECMG 0409/0411 01010KT BECMG 0417/0419
       VRB05KT=
```

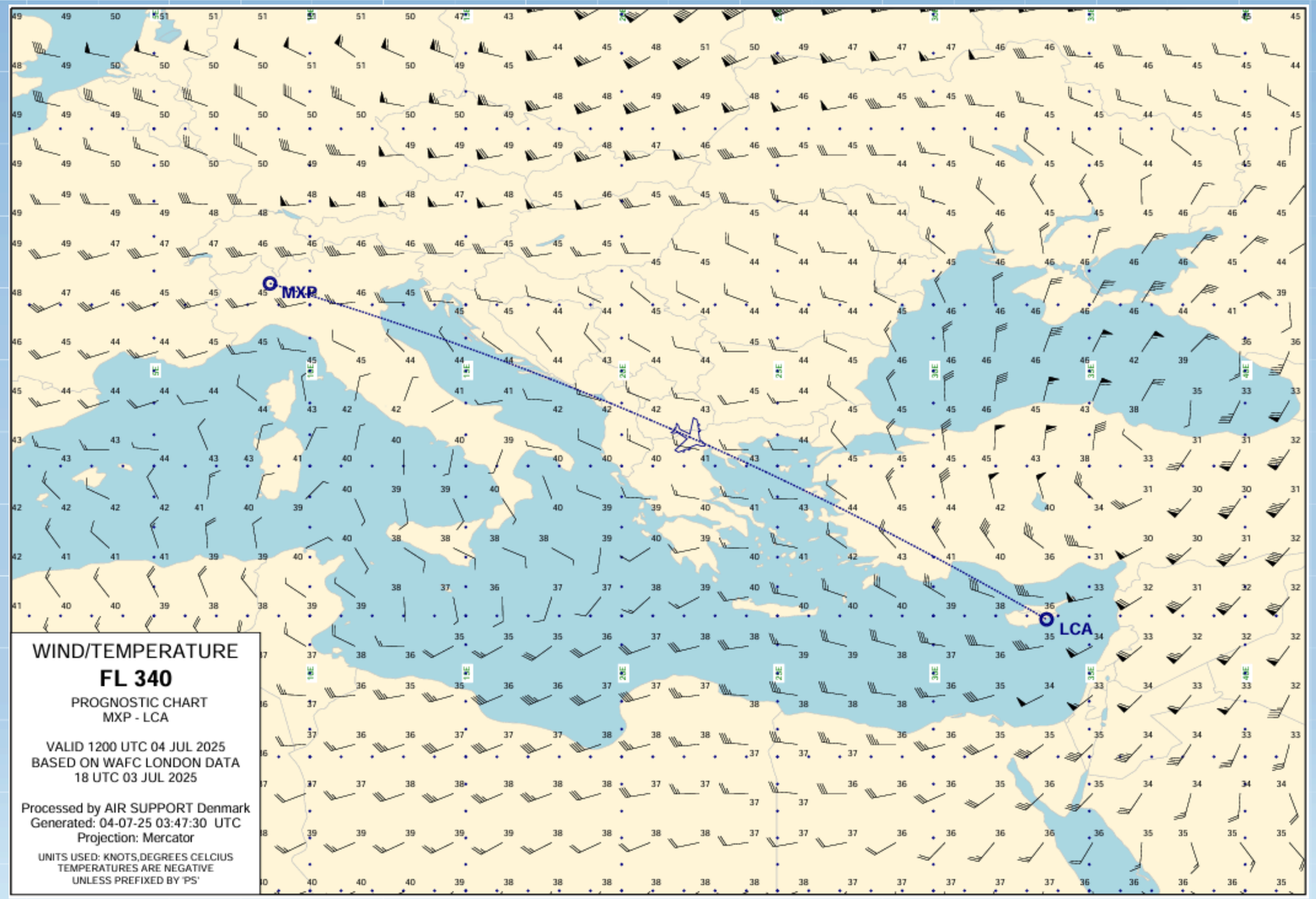
Airport LGAV - ATH - ATHENS/ELEFTHERIOS VENIZELOS RWY 03L 03R 21L 21R

```
METAR  040320Z  36010KT CAVOK 26/06 Q1016 NOSIG=
TAF    032300Z  0400/0424 35012KT CAVOK BECMG 0406/0408 04020KT TEMPO 0410/0416
       04020G30KT=
```

TASKS

- -Check both airports
- -Select route and FL
- -Fuel consumption and required
- -Alternate airports
- -Adequate airports
- -Weather and NOTAMs
- -Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

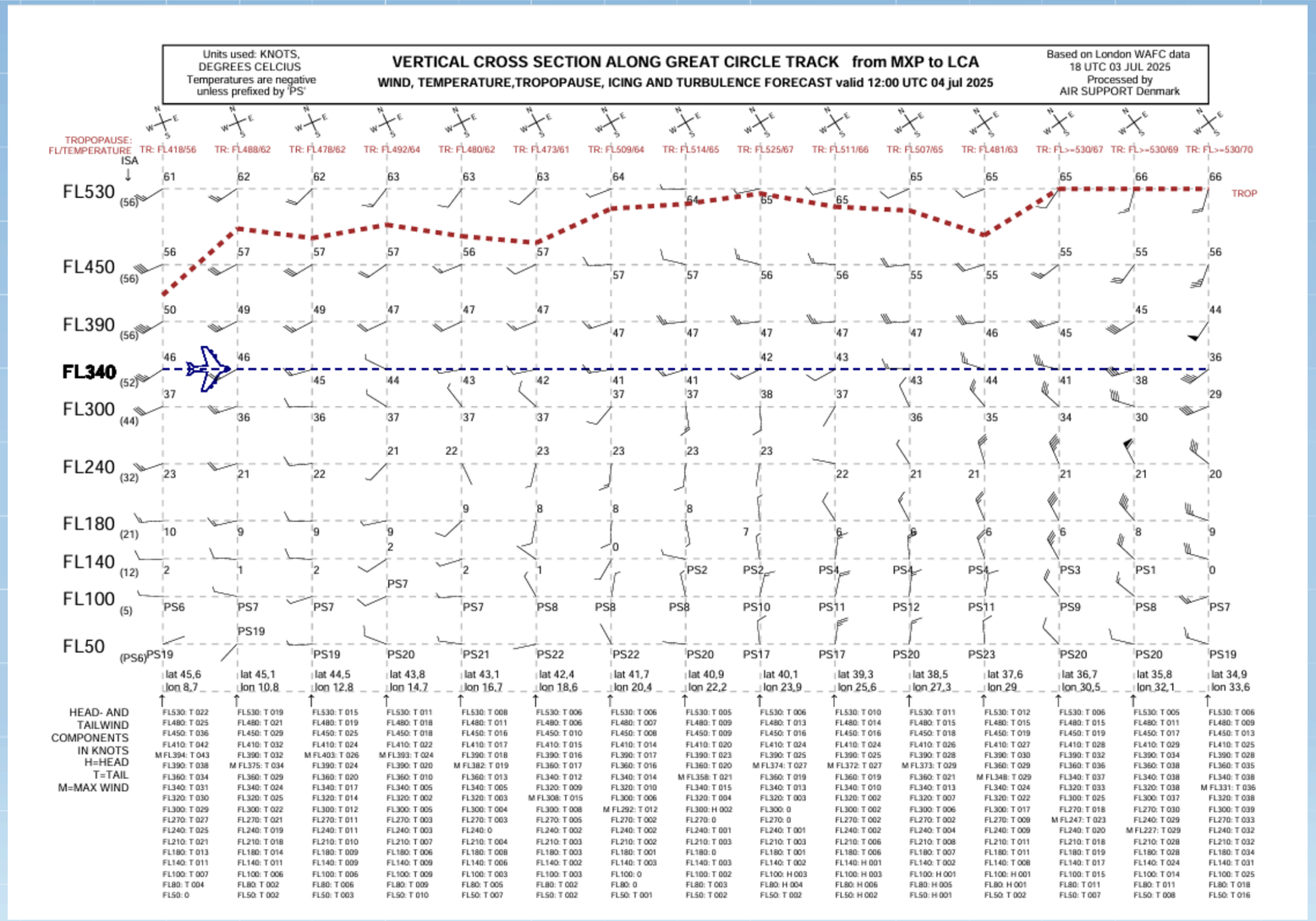
-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

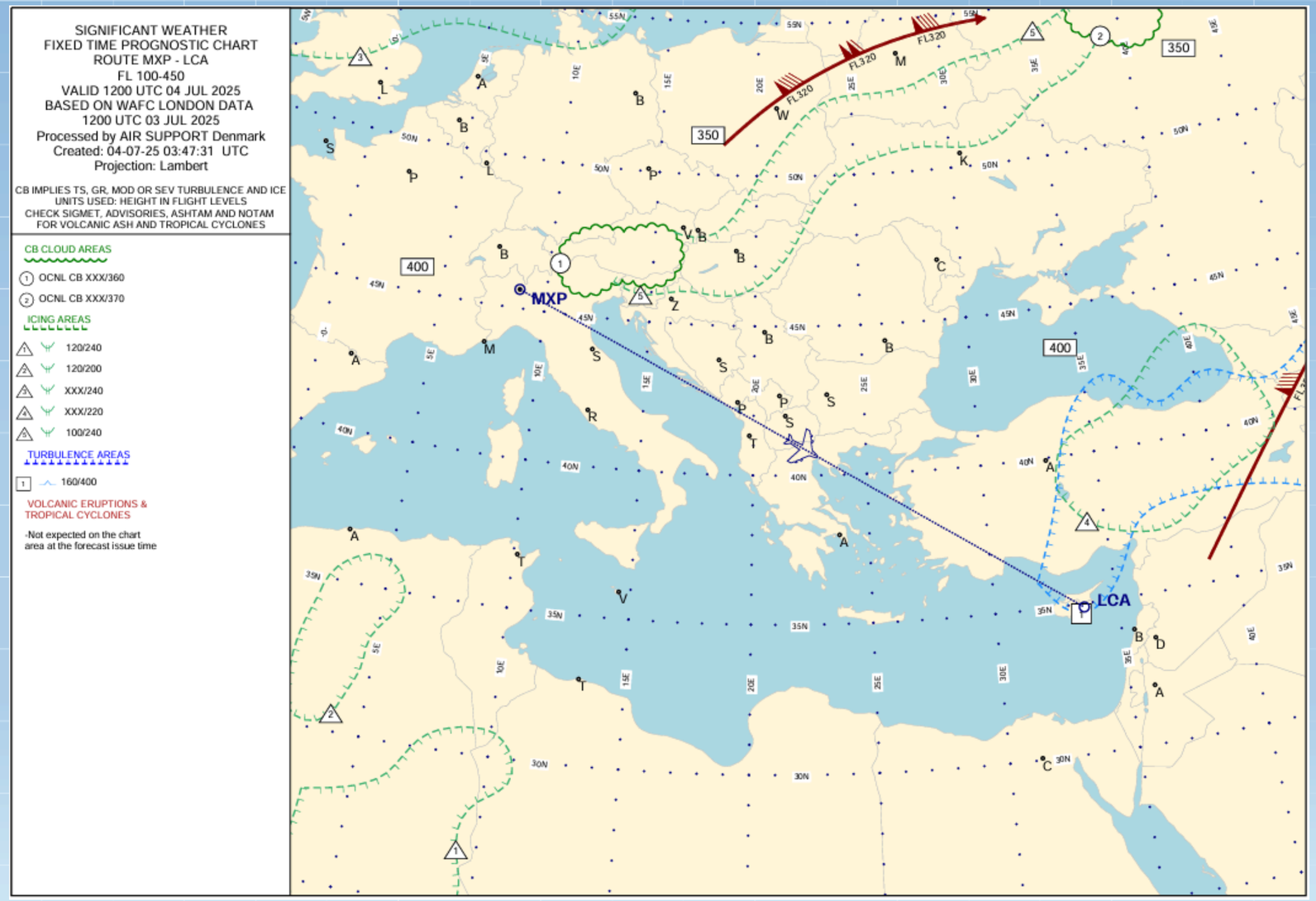
-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:



TASKS

-Check both airports

-Select route and FL

-Fuel consumption and required

-Alternate airports

-Adequate airports

-Weather and NOTAMs

-Etc.

TO DO:

Airport LIMC

```
|#1|--------------------------------------------------------------------------------
A3793/25 NOTAMN                                                    [sid | -33]
Q) LIMM/QPDTT/I/BO/A/000/999/4538N00843E005
A) LIMC B) 2508070000 C) 2508202359
E) TRIGGER NOTAM-PERM AIP AIRAC AMDT 8/25 EFFECTIVE DATE 07 AUG 2025
MILANO/MALPENSA AD
REVIEW OF SID
AIRAC IS POSTED AND AVBL ON WEBSITE WWW.ENAV.IT

|#2|--------------------------------------------------------------------------------
A3169/25 NOTAMN                                                     [ad | -5]
Q) LIMM/QFATT/IV/BO/A/000/999/4538N00843E005
A) LIMC B) 2507100000 C) 2507232359
E) TRIGGER NOTAM-PERM AIP AIRAC AMDT 7/25 EFFECTIVE DATE 10 JUL 2025
MILANO/MALPENSA AD
TABLE AD 2.20 ''''LOCAL AERODROME REGULATIONS'''' UPDATED
AIRAC IS POSTED AND AVBL ON WEBSITE WWW.ENAV.IT

|#3|--------------------------------------------------------------------------------
A3972/25 NOTAMN                                          [obst | NEW TODAY]
Q) LIMM/QOBCE/IV/M/A/000/999/4538N00843E005
A) LIMC B) 2507042200 C) 2507060100
D) DAILY 2200-0100
E) NEW OBSTACLE ERECTED MOBILE CRANE PENETRATING
INNER HORIZONTAL SFC AND TRANSITIONAL SFC RWY 17L/35R
PSN COORD (WGS-84): 453845.9N 0084328.7E
MAX ELEV AGL 67.3M/220.8FT MAX ELEV AMSL 297.3M/975.4FT
ROTATING JIB 67.3M/220.8FT
ICAO SIGNALS PROVIDED
REF AIP AD2 LIMC 3-5

|#4|--------------------------------------------------------------------------------
A3896/25 NOTAMN                                          [acft stand | 1]
Q) LIMM/QMPLC/IV/BO/A/000/999/4538N00843E005
A) LIMC B) 2507020500 C) 2507071800
E) ACFT STANDS 751, 752 AND 753 CLSD DUE TO WIP
RMK: IACAO SIGNALS PROVIDED
REF AIP AD 2 LIMC 2-11
```

TASKS

- -Check both airports
- -Select route and FL
- -Fuel consumption and required
- -Alternate airports
- -Adequate airports
- -Weather and NOTAMs
- -Etc.

TO DO:

Airport LIMC

```
|#1|--------------------------------------------------------------------------------
A3793/25 NOTAMN                                                    [sid | -33]
Q) LIMM/QPDTT/I/BO/A/000/999/4538N00843E005
A) LIMC B) 2508070000 C) 2508202359
E) TRIGGER NOTAM-PERM AIP AIRAC AMDT 8/25 EFFECTIVE DATE 07 AUG 2025
MILANO/MALPENSA AD
REVIEW OF SID
AIRAC IS POSTED AND AVBL ON WEBSITE WWW.ENAV.IT

|#2|--------------------------------------------------------------------------------
A3169/25 NOTAMN                                                     [ad | -5]
Q) LIMM/QFATT/IV/BO/A/000/999/4538N00843E005
A) LIMC B) 2507100000 C) 2507232359
E) TRIGGER NOTAM-PERM AIP AIRAC AMDT 7/25 EFFECTIVE DATE 10 JUL 2025
MILANO/MALPENSA AD
TABLE AD 2.20 ''''LOCAL AERODROME REGULATIONS'''' UPDATED
AIRAC IS POSTED AND AVBL ON WEBSITE WWW.ENAV.IT

|#3|--------------------------------------------------------------------------------
A3972/25 NOTAMN                                          [obst | NEW TODAY]
Q) LIMM/QOBCE/IV/M/A/000/999/4538N00843E005
A) LIMC B) 2507042200 C) 2507060100
D) DAILY 2200-0100
E) NEW OBSTACLE ERECTED MOBILE CRANE PENETRATING
INNER HORIZONTAL SFC AND TRANSITIONAL SFC RWY 17L/35R
PSN COORD (WGS-84): 453845.9N 0084328.7E
MAX ELEV AGL 67.3M/220.8FT MAX ELEV AMSL 297.3M/975.4FT
ROTATING JIB 67.3M/220.8FT
ICAO SIGNALS PROVIDED
REF AIP AD2 LIMC 3-5

|#4|--------------------------------------------------------------------------------
A3896/25 NOTAMN                                          [acft stand | 1]
Q) LIMM/QMPLC/IV/BO/A/000/999/4538N00843E005
A) LIMC B) 2507020500 C) 2507071800
E) ACFT STANDS 751, 752 AND 753 CLSD DUE TO WIP
RMK: IACAO SIGNALS PROVIDED
REF AIP AD 2 LIMC 2-11
```

TASKS

- -Check both airports
- -Select route and FL
- -Fuel consumption and required
- -Alternate airports
- -Adequate airports
- -Weather and NOTAMs
- -Etc.

LAST

MINUTE

CHANGES

LAST MINUTE CHANGES

A CERTAIN TYPE OF LAST MINUTE CHANGE:

THE PASSENGERS

A CERTAIN TYPE OF LAST MINUTE CHANGE:

THE PASSENGERS

WITH

python™

SUDDEN CHANGE IN NUMBER OF PASSENGERS:

LAST MINUTE PASSENGERS

COMMON

- This more common than people realize, and solving it is difficult.

PREDICTION*

- Might depend on the day of the week, the time of the day, a special event...

* This is not possible most of the time due to the enormous amount of factors, and its complexity goes beyond the capability of any possible... Are you still reading? Then, read Brandon Sanderson, please. I absolutely adore his books - and also follow me on Twitter @tapacubosfumar. I just complain about stuff.

A PYTHON LIBRARY:

pandas

POWERFUL

DataFrames make it possible to manipulate easily big chunks of data

MORE LIBRARIES

DATETIME

To work with dates, times, time zones...

MATPLOTLIB

To help visualize data with charts and graphs

NUMPY

The quintessencial Python library

DATAFRAME

	FLIGHT NUMBER	DEP	ARR	DATE	TIME	WEEKDAY	PAX TRUE	PAX FALSE	PAX TOTAL
0	HT0928	TFN	LPA	2025-01-01	05:20	2	70	1	71
1	HT0922	LPA	TFN	2025-01-01	06:00	2	42	8	50
2	HT1650	FCO	CAG	2025-01-01	06:15	2	71	63	134
3	HT7614	CAG	FCO	2025-01-01	07:30	2	118	16	134
4	HT7132	MXP	SSH	2025-01-01	07:30	2	436	14	450
5	HT0928	TFN	LPA	2025-01-01	12:00	2	33	24	57
6	HT1650	FCO	CAG	2025-01-01	12:30	2	67	58	125
7	HT0922	LPA	TFN	2025-01-01	12:40	2	32	26	58
8	HT7614	CAG	FCO	2025-01-01	13:45	2	69	58	127
9	HT1650	FCO	CAG	2025-01-01	16:45	2	73	5	78

WAS THE TICKET BOUGHT ONLINE?

WAS THE TICKET BOUGHT ONLINE?

YES / TRUE

DATA WE HAD WHILE PLANNING THE FLIGHT

NO / FALSE

DATA WE DID NOT HAVE. PROBLEMS!

```
    # Loading the .CSV into a variable
df = pd.read_csv("Data.csv")


    # List to store the filters that we might apply
filter_bank = []

while input("Would you like to run the script? (yes/no): ").lower() == "yes":

    # Filter by flight number
    yes_no_fn = input("Would you like to filter by flight number? (yes/no): ").lower()
    if yes_no_fn == "yes":
        fn_req = int(input("Please select a flight number: "))
        try:
            if fn_req in df["FLIGHT NUMBER"].unique():
                filter_bank.append(("FLIGHT NUMBER", (fn_req)))
            else:
                print("The flight number selected does not exist.")
        except ValueError:
            print("Invalid flight number. Please enter a numeric value.")

    # Filter by departure
    yes_no_dep = input("Would you like to filter by departure? (yes/no): ").lower()

    if yes_no_dep == "yes":
        dep_req = str(input("Please select a departure: "))
        try:
            if dep_req in df["DEP"].unique():
                filter_bank.append(("DEP", (dep_req)))
            else:
                print("Airport of departure does not exist.")
        except ValueError:
            print("Invalid departure. Please enter a three letter code.")

    # Filter by destination
    yes_no_dest = input("Would you like to filter by destination? (yes/no): ").lower()

    if yes_no_dest == "yes":
        dest_req = str(input("Please select an arrival: "))
        try:
            if dest_req in df["ARR"].unique():
                filter_bank.append(("ARR", (dest_req)))
            else:
                print("Airport of arrival does not exist.")
        except ValueError:
            print("Invalid arrival. Please enter a three letter code.")

    # Filter by date
    yes_no_date = input("Would you like to filter by date? (yes/no): ").lower()

    if yes_no_date == "yes":
        date_req = input("Which date would you like to select (YYYY-MM-DD): ")
        try:
            if yes_no_fn == "yes":
                if date_req in list(df[df["Flight number"] == fn_req]["Date"]):
                    filter_bank.append(("Date", (date_req)))
                else:
                    print("There is no flight %d departing at %s." %(fn_req, date_req))
            else:
                filter_bank.append(("DATE", (date_req)))
        except ValueError:
            print("Invalid date format. Please use YYYY-MM-DD format.")

    # Filter by day of the week
    yes_no_week = input("Would you like to filter by day of the week? (yes/no): ").lower()
    if yes_no_week == "yes":
        day_req = int(input("Which day would you like to select (0 - 6): "))
        try:
            filter_bank.append(("WEEKDAY", (day_req)))
        except ValueError:
            print("Invalid date format. Please use 0-6.")

    # Filter by time
    yes_no_time = input("Would you like to filter by time? (yes/no): ").lower()
    if yes_no_time == "yes":
        time_req = input("Which time would you like to select (HH:MM): ")
        try:
            filter_bank.append(("TIME", (time_req)))
        except ValueError:
            print("Invalid time format. Please use HH:MM.")

    print("Current filters:", filter_bank)

# Apply filters to the DataFrame creating a copy
filtered_df = df.copy()
```

```
    # Loading the .CSV into a variable
df = pd.read_csv("Data.csv")


    # List to store the filters that we might apply
filter_bank = []

while input("Would you like to run the script? (yes/no): ").lower() == "yes":

    # Filter by flight number
    yes_no_fn = input("Would you like to filter by flight number? (yes/no): ").lower()
    if yes_no_fn == "yes":
        fn_req = int(input("Please select a flight number: "))
        try:
            if fn_req in df["FLIGHT NUMBER"].unique():
                filter_bank.append(("FLIGHT NUMBER", (fn_req)))
            else:
                print("The flight number selected does not exist.")
        except ValueError:
            print("Invalid flight number.

    # Filter by departure
    yes_no_dep = input("Would you like to

    if yes_no_dep == "yes":
        dep_req = str(input("Please select
        try:
            if dep_req in df["DEP"].unique():
                filter_bank.append(("DEP", (dep_req)))
            else:
                print("Airport of departure does not exist.")
        except ValueError:
            print("Invalid departure. Please enter a three letter code.")

    # Filter by destination
    yes_no_dest = input("Would you like to filter by destination? (yes/no): ").lower()

    if yes_no_dest == "yes":
        dest_req = str(input("Please select an arrival: "))
        try:
            if dest_req in df["ARR"].unique():
                filter_bank.append(("ARR", (dest_req)))
            else:
                print("Airport of arrival does not exist.")
        except ValueError:
            print("Invalid arrival. Please enter a three letter code.")
```


```
    # Filter by date
    yes_no_date = input("Would you like to filter by date? (yes/no): ").lower()

    if yes_no_date == "yes":
        date_req = input("Which date would you like to select (YYYY-MM-DD): ")
        try:
            if yes_no_fn == "yes":
                if date_req in list(df[df["Flight number"] == fn_req]["Date"]):
                    filter_bank.append(("Date", (date_req)))
                else:
                    print("There is no flight %d departing at %s." %(fn_req, date_req))
            else:
                filter_bank.append(("DATE", (date_req)))
        except ValueError:
            print("Invalid date format. Please use YYYY-MM-DD format.")


                                                    ilter by day of the week? (yes/no): ").lower()

                                                     you like to select (0 - 6): "))

                                                    day_req)))
            except ValueError:
                print("Invalid date format. Please use 0-6.")

    # Filter by time
    yes_no_time = input("Would you like to filter by time? (yes/no): ").lower()
    if yes_no_time == "yes":
        time_req = input("Which time would you like to select (HH:MM): ")
        try:
            filter_bank.append(("TIME", (time_req)))
        except ValueError:
            print("Invalid time format. Please use HH:MM.")

    print("Current filters:", filter_bank)

# Apply filters to the DataFrame creating a copy
filtered_df = df.copy()
```

DATAFRAME

	FLIGHT NUMBER	DEP	ARR	DATE	TIME	WEEKDAY	PAX TRUE	PAX FALSE	PAX TOTAL
0	HT0928	TFN	LPA	2025-01-01	05:20	2	70	1	71
1	HT0922	LPA	TFN	2025-01-01	06:00	2	42	8	50
2	HT1650	FCO	CAG	2025-01-01	06:15	2	71	63	134
3	HT7614	CAG	FCO	2025-01-01	07:30	2	118	16	134
4	HT7132	MXP	SSH	2025-01-01	07:30	2	436	14	450
5	HT0928	TFN	LPA	2025-01-01	12:00	2	33	24	57
6	HT1650	FCO	CAG	2025-01-01	12:30	2	67	58	125
7	HT0922	LPA	TFN	2025-01-01	12:40	2	32	26	58
8	HT7614	CAG	FCO	2025-01-01	13:45	2	69	58	127
9	HT1650	FCO	CAG	2025-01-01	16:45	2	73	5	78

INPUTS

1ST ELEMENT

```
Current filters: [('DEP', 'TFN'), ('WEEKDAY', 4)]
```

2ND ELEMENT

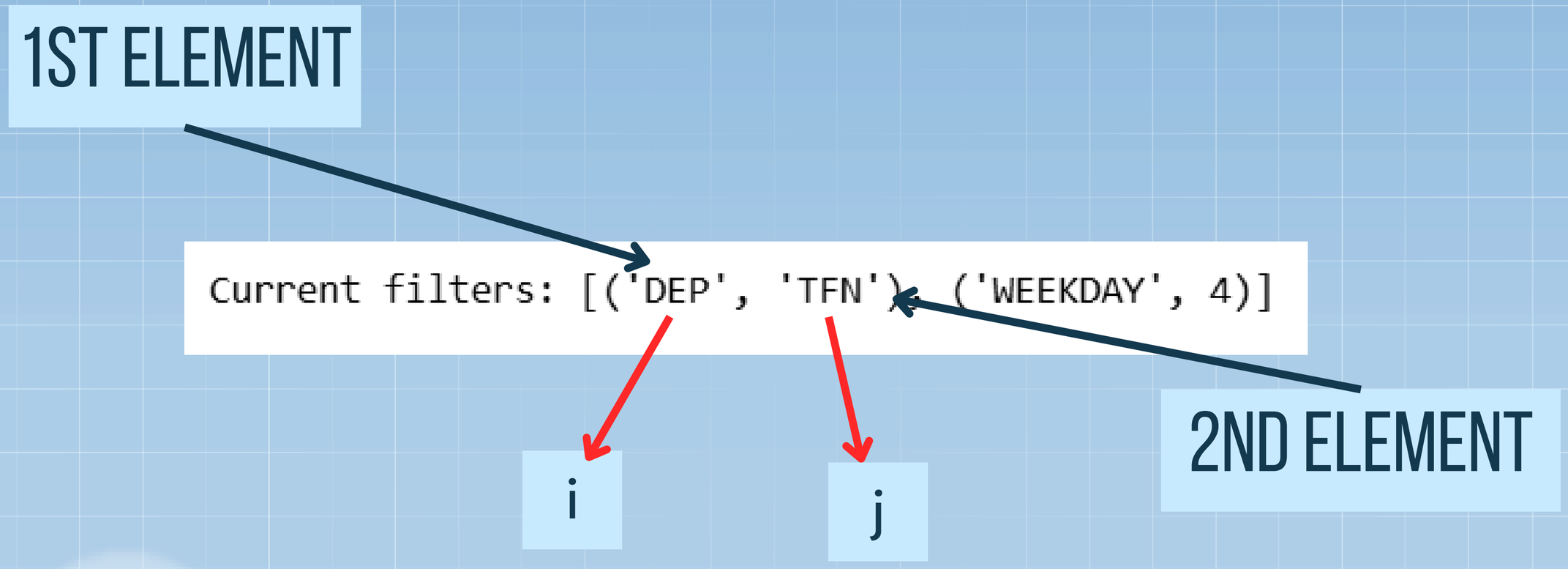
THE **CORE** OF THE SCRIPT

```
# ---------------------- CREATING THE FILTERED DATAFRAME ------------------------

for i, j in filter_bank:
    filtered_df = filtered_df[filtered_df[i] == j]
```

THE **CORE** OF THE SCRIPT

```
# ---------------------- CREATING THE FILTERED DATAFRAME ------------------------

for i, j in filter_bank:
    filtered_df = filtered_df[filtered_df[i] == j]
```

INPUTS


1ST ELEMENT
Current filters: [('DEP', 'TFN'), ('WEEKDAY', 4)]
i
j
2ND ELEMENT

Filtered DataFrame:

	FLIGHT NUMBER	DEP	ARR	DATE	TIME	WEEKDAY	PAX TRUE	PAX FALSE	PAX TOTAL
0	HT0928	TFN	LPA	2025-01-03	05:20	4	52	18	70
1	HT0928	TFN	LPA	2025-01-03	12:00	4	33	37	70
2	HT3677	TFN	BCN	2025-01-03	12:30	4	150	24	174
3	HT0928	TFN	LPA	2025-01-03	19:55	4	28	25	53
4	HT0928	TFN	LPA	2025-01-10	05:20	4	44	16	60
5	HT0928	TFN	LPA	2025-01-10	12:00	4	30	30	60
6	HT3677	TFN	BCN	2025-01-10	12:30	4	111	11	122
7	HT0928	TFN	LPA	2025-01-10	19:55	4	36	25	61
8	HT0928	TFN	LPA	2025-01-17	05:20	4	47	2	49
9	HT0928	TFN	LPA	2025-01-17	12:00	4	31	30	61
10	HT3677	TFN	BCN	2025-01-17	12:30	4	147	14	161
11	HT0928	TFN	LPA	2025-01-17	19:55	4	28	21	49

ANALYSING THE OUTCOME

```
# ------------------------------ CALCULATING % OF LMP ------------------------------ #

LMP_per = sum(filtered_df["PAX FALSE"]) / sum(filtered_df["PAX TRUE"]) * 100

# ---------------------------------------------------------------------------------- #
```

Selected flights have suffered a last minute increase of 36.39% on average.

ANY OTHER FUNCTION WE WANT/NEED

BUSINESS CONTACT INFO

LINKEDIN

www.linkedin.com
/in/joaquin-
rayado-
menjon-627422313

EMAIL

j.rayado@airhorizont.com

PHONE

+356 9904 3996

LET ME KNOW IF YOU HAVE ANY QUESTIONS

JoaquinRayado / FlyingFree