In [47]:

```
import csv
import json
import os
import warnings

import geopy.distance
import numpy as np
import pandas as pd
import requests as req
import tqdm
```

In [48]:

```
warnings.filterwarnings("ignore")
my_path = os.getcwd()
```

In [3]:

```
# file with metro coordinates
df_metro_full = pd.read_excel(rf"{my_path}\metro_coordinates.xlsx")

df_metro = df_metro_full.drop(["id", "lat", "lon", "station_name"], axis=1)
```

In [4]:

```
# file with flats' coordinates
df_flats_full = pd.read_excel(rf"{my_path}\maket_coordinates_Anton.xlsx")

df_flats = df_flats_full.drop(["address_abbr"], axis=1)
df_flats = df_flats.drop_duplicates(ignore_index=True)
```

In [51]:

```
# lists of strings -> lists of tuples
li_metro = list(df_metro["coordinates"].values)
li_of_lists_metro = [elem.split(",") for elem in li_metro]
li_metro = [tuple(float(elem_2) for elem_2 in elem) for elem in li_of_lists_metro
]

li_flats = list(df_flats["coord"].values)
li_of_lists_flats = [elem.split(",") for elem in li_flats]
li_flats = [tuple(float(elem_2) for elem_2 in elem) for elem in li_of_lists_flats
]
```

Driving distance

In [17]:

```
with open(rf"{my_path}\driving_distance.csv", "a", encoding="utf-8", newline="")
as ouf:
    writer = csv.writer(ouf, delimiter="$", quotechar="|", quoting=csv.QUOTE_MINI
MAL)
    writer.writerow(
        (
            "flat_coord",
            "metro_coord",
            "driving_distance"
```

```
        )
    )


def distance_API(orig_coord: tuple, dest_coord: tuple) -> float:

    lon_1 = orig_coord[1]
    lat_1 = orig_coord[0]

    lon_2 = dest_coord[1]
    lat_2 = dest_coord[0]

    r = req.get(
        f"http://router.project-
osrm.org/route/v1/driving/{lon_1},{lat_1};{lon_2},{lat_2}?overview=false"
        ""
    )
    routes = json.loads(r.content)
    route_1 = routes.get("routes")[0]["distance"]
    return route_1 / 1000


for flat in tqdm.tqdm(li_flats):
    for metro in li_metro:
        distance = distance_API(flat, metro)
        data = (flat, metro, distance)
        with open(rf"{my_path}\driving_distance.csv", "a", encoding="utf-8", newl
ine="") as ouf:
            writer = csv.writer(
                ouf,
                delimiter="$",
                quotechar="|",
                quoting=csv.QUOTE_MINIMAL,
            )
            writer.writerow(data)
```

```
100%|██████████| 3/3 [01:49<00:00, 36.55s/it]
```

In [35]:

```
df_calculated = pd.read_csv(rf"{my_path}\driving_distance.csv", delimiter="$")
li_indices = []

for flat in li_flats:
    df_small = df_calculated[df_calculated["flat_coord"]==str(flat)]
    min_index = df_small[['driving_distance']].idxmin()[0]
    li_indices.append(min_index)
```

In [ ]:

```
df_result = df_calculated.iloc[li_indices, :]
df_result['metro_coord'] = df_result['metro_coord'].str.replace("\(", "").str.rep
lace("\)", "")

df_result = pd.merge(df_result, df_metro_full[["coordinates", "station_name"]].s
et_index('coordinates'), left_on='metro_coord', right_index=True)
```

Geo distance

In [52]:

```
with open(rf"{my_path}\geo_distance.csv", "w", encoding="utf-8", newline="") as
```

```
ouf:
    writer = csv.writer(ouf, delimiter="$", quotechar="|", quoting=csv.QUOTE_MINI
MAL)
    writer.writerow(
        (
            "flat_coord",
            "metro_coord",
            "geo_distance"
        )
    )


for flat in tqdm.tqdm(li_flats):
    for metro in li_metro:
        distance = geopy.distance.geodesic(flat, metro).km
        data = (flat, metro, distance)
        with open(rf"{my_path}\geo_distance.csv", "a", encoding="utf-8", newline=
"") as ouf:
            writer = csv.writer(
                ouf,
                delimiter="$",
                quotechar="|",
                quoting=csv.QUOTE_MINIMAL,
            )
            writer.writerow(data)
```

```
100%|██████████| 568/568 [01:16<00:00,  7.38it/s]
```

In [53]:

```
df_calculated2 = pd.read_csv(rf"{my_path}\geo_distance.csv", delimiter="$")
li_indices2 = []

for flat in li_flats:
    df_small = df_calculated2[df_calculated2["flat_coord"]==str(flat)]
    min_index = df_small[['geo_distance']].idxmin()[0]
    li_indices2.append(min_index)
```

In [54]:

```
df_result2 = df_calculated2.iloc[li_indices2, :]
df_result2['metro_coord'] = df_result2['metro_coord'].str.replace("\(", "").str.r
eplace("\)", "")

df_result2 = pd.merge(df_result2, df_metro_full[["coordinates", "station_name"]]
.set_index('coordinates'), left_on='metro_coord', right_index=True)
```

Union 2 types of distance

In [50]:

```
df_all_dist = pd.merge(df_result, df_result2, on="flat_coord", suffixes=("_drivin
g","_geo"))
df_all_dist = df_all_dist.drop(labels=["metro_coord_driving",
"metro_coord_geo"], axis=1)
df_all_dist['flat_coord'] = df_all_dist['flat_coord'].str.replace("\(", "").str.r
eplace("\)", "")

df_all_dist.to_excel(rf"{my_path}\all_distances.xlsx", index=False)
```