Vodafon домашнє завдання .

Як використовувати код:

1. **Підготувати файл з координатами**: Розмістіть текстовий файл із координатами у форматі [longitude, latitude] за вашим шляхом.
2. **Залежності**: перевірте чи встановлені потібні Python-бібліотеки: psycopg2 для роботи з PostgreSQL, folium для візуалізації карти.
3. **Налаштувати підключення до бази даних**: У параметрах connection_params вкажіть ваші дані для підключення до PostgreSQL.
   ps: впевниться, що PostgreSQL підтримує розширення PostGIS.
4. **Запустити код**: Після виконання скрипта буде створено базу даних, таблицю, завантажено координати, а також збережено HTML-карту.
5. **Переглянути карту**: Відкрийте html файл у браузері для перегляду.

1 та 2.

Задача:Знайти координати кордону України та занести їх у базу даних Oracle/PostgreSQL/MySQL. Графічно відобразити кордон у середовищі Python або на веб-інтерфейсі за допомогою бібліотеки Leaflet.

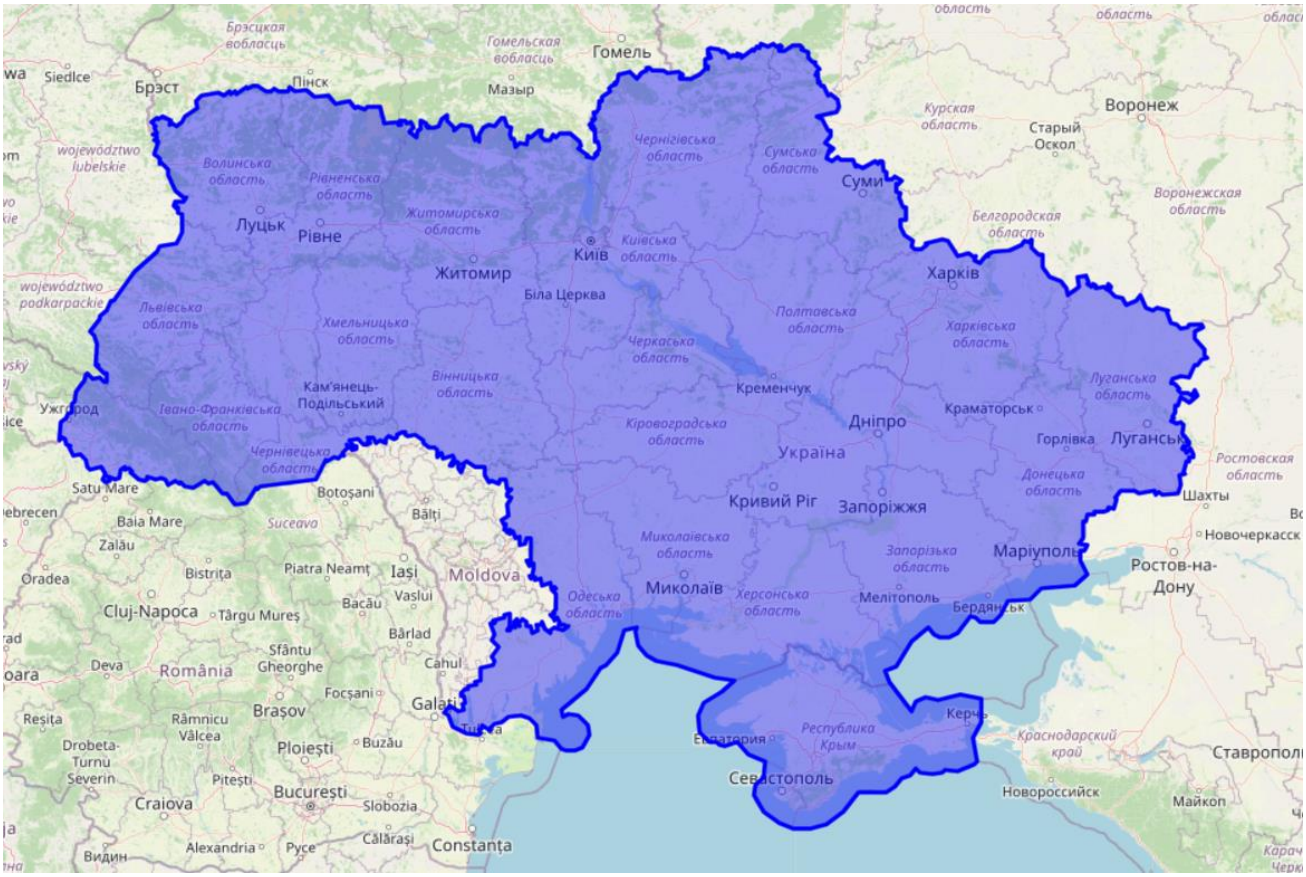
Розбір коду**:**

1. **Зчитування координат із файлу**: Функція read_coordinates_from_file обробляє текстовий файл, який містить координати у форматі [longitude, latitude]. Координати обробляються і перетворюються на список пар [(latitude, longitude), ...].
2. **Створення бази даних та підключення PostGIS**: Код перевіряє, чи існує база даних, і створює її у разі відсутності. Також активується розширення PostGIS для роботи з геометричними даними.**Ключові функції:**
   - create_database_if_not_exists: перевіряє існування бази даних та створює її.
   - Створення таблиці ukraine_borders для збереження географічних точок із використанням типу GEOMETRY(Point, 4326).
3. **Додавання координат у базу**: Координати зчитуються з файлу та додаються в таблицю ukraine_borders за допомогою ST_SetSRID(ST_MakePoint(%s, %s), 4326).
4. **Витягування координат із бази**: Дані витягуються у вигляді списку координат для подальшого використання. Координати витягуються у форматі longitude, latitude.
5. **Візуалізація кордону за допомогою Folium**: Бібліотека Folium використовується для створення HTML-карти. Координати додаються як полігон, що відображає кордон України.**Основні кроки:**
   - Обчислення центру карти на основі середніх значень широти та довготи.
   - Створення полігону для відображення кордону.
   - Збереження карти у файл ukraine_borders_map.html.

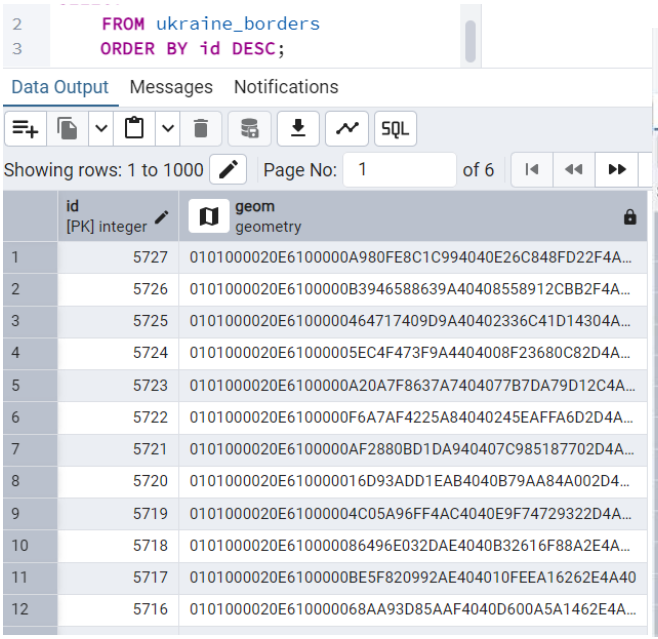
Консольний вивід:

```
База даних 'coordinates_uk_db' вже існує.
PostGIS розширення підключено.
Таблиця 'ukraine_borders' створена .
Координати додано до таблиці.
Карта збережена у файл 'ukraine_borders_map.html'. Відкрийте файл у браузері для перегляду.
```

HTML-карта:



Вигрузка з ukraine_borders:

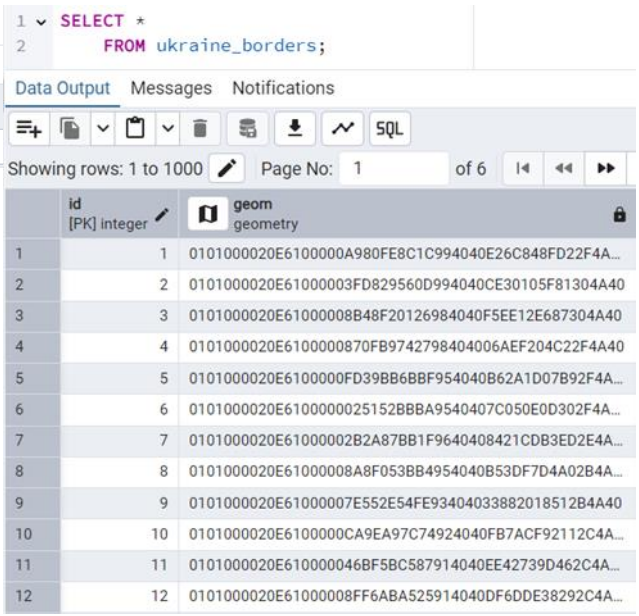


```
    FROM ukraine_borders
    ORDER BY id DESC;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1000 | Page No: 1 of 6

	id [PK] integer	geom geometry
1	5727	0101000020E6100000A980FE8C1C994040E26C848FD22F4A...
2	5726	0101000020E6100000B3946588639A40408558912CBB2F4A...
3	5725	0101000020E6100000464717409D9A40402336C41D14304A...
4	5724	0101000020E61000005EC4F473F9A4404008F23680C82D4A...
5	5723	0101000020E6100000A20A7F8637A7404077B7DA79D12C4A...
6	5722	0101000020E6100000F6A7AF4225A84040245EAFFA6D2D4A...
7	5721	0101000020E6100000AF2880BD1DA940407C985187702D4A...
8	5720	0101000020E610000016D93ADD1EAB4040B79AA84A002D4...
9	5719	0101000020E61000004C05A96FF4AC4040E9F74729322D4A...
10	5718	0101000020E6100000086496E032DAE4040B32616F88A2E4A...
11	5717	0101000020E6100000BE5F820992AE404010FEEA16262E4A40
12	5716	0101000020E610000068AA93D85AAF4040D600A5A1462E4A...



```
SELECT *
    FROM ukraine_borders;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1000 | Page No: 1 of 6

	id [PK] integer	geom geometry
1	1	0101000020E6100000A980FE8C1C994040E26C848FD22F4A...
2	2	0101000020E61000003FD829560D994040CE30105F81304A40
3	3	0101000020E61000008B48F20126984040F5EE12E687304A40
4	4	0101000020E6100000870FB9742798404006AEF204C22F4A40
5	5	0101000020E6100000FD39BB6BBF954040B62A1D07B92F4A...
6	6	0101000020E6100000025152BBBA9540407C050E0D302F4A...
7	7	0101000020E61000002B2A87BB1F9640408421CDB3ED2E4A...
8	8	0101000020E61000008A8F053BB4954040B53DF7D4A02B4A...
9	9	0101000020E61000007E552E54FE93404033882018512B4A40
10	10	0101000020E6100000CA9EA97C74924040FB7ACF92112C4A...
11	11	0101000020E6100000046BF5BC587914040EE42739D462C4A...
12	12	0101000020E61000008FF6ABA525914040DF6DDE38292C4A...

Додаток:Було прийняте рішення вигружати саме з текстового файлу .Можливо в нього було ліпше зберігати у форматі geojson для оптимізації. Такі варіації коду були.Проте хотілося зробити код без додаткових бібліотек окрім потрібної для візуалізації та підключення до postgres. Також було прийняте рішення зберігати точки не в форматі двох стовпців широти та довготи, а використати можливості postgis і зберігати у вигляді об’єкта точки.

3 та 4.

Задача: Запропонувати алгоритм, який розбиватиме карту України на однакові квадрати (сторона ~1 км, можна і більше, якщо не вистачає ресурсів на ноутбуці). Зберегти вершини сформованих квадратів у базу даних Oracle/PostgreSQL/MySQL.
PS: з 1 км дуже довгі обрахунки і виходить опісля завеликий html файл, тож працюю з 10 км. Подальше пояснення в додатку.

Розбір коду та алгоритму:

1. **Підключення до бази даних:**Створюється підключення до бази даних coordinates_uk_db, яка була раніше налаштована для зберігання географічних даних.
2. **Створення таблиці для збереження точок**: Таблиця square_vertices_10 створюється для зберігання вершин квадратів. Поля таблиці: id — унікальний ідентифікатор; diagonal_id і vertical_id — координати точки в сітці за діагональним і вертикальним індексами. ;geom — геометричний тип (точка) для роботи з PostGIS.
3. **Завантаження даних про кордони України**: Координати кордонів України (полігон) вже збережені в таблиці ukraine_borders. Дані витягуються та формуються у вигляді списку координат, які будуть використані для побудови полігону. Якщо таблиця ukraine_borders порожня, алгоритм завершує виконання.

4. **Побудова полігону України**:Полігон створюється на основі витягнутих координат і записується в таблицю ukraine_polygon. Це дозволяє оптимізувати перевірку точок на входження до полігону.Так як для такої кількості ітерацій створювати кожен раз полігон у «повітрі» дуже затратно по пам’яті.
5. **Визначення меж полігону**: Розраховуються мінімальні та максимальні значення широти й довготи для полігону України. Ці межі використовуються для створення прямокутної області, що охоплює полігон.
6. **Розбиття прямокутної області на сітку**:Визначається крок сітки у градусах (як довжина сторони квадрата в градусах), який залежить від заданої довжини сторони квадрата (10 км у цьому випадку).Формули переведення:
   Широта: degree_per_km_lat=довжина_км/(111×k) , k=1.5 для корекції реальних метричних розрахунків.
   Довгота: degree_per_km_lon=довжина_км/111
   Кількість кроків у кожному напрямку розраховується залежно від розміру області.
7. **Генерація сітки точок**:Створюється регулярна сітка точок на основі розрахованих меж і кроків.Для кожної точки виконується перевірка: чи входить вона у межі полігону України. Ця перевірка здійснюється за допомогою функції ST_Contains PostGIS.
8. **Збереження валідних точок у базу даних**:Точки, які потрапили в межі полігону, зберігаються у таблицю square_vertices_10.

Консольний вивід:

```
Таблиця 'square_vertices_10' створена.
Таблиця 'ukraine_polygon' створена.
Полігон України додано до таблиці 'ukraine_polygon'.
У базу даних додано 14498 точок, які входять у межі полігону України.
```

Вигрузка з square_vertices_10:

```
SELECT *
    FROM square_vertices_10;
```

Data Output | Messages | Notifications — Showing rows: 1 to 1000

	id [PK] integer	diagonal_id integer	vertical_id integer	geom geometry
1	1	1	127	0101000020E610000027C5DE0321C
2	2	1	128	0101000020E6100000E0155216A9D
3	3	1	129	0101000020E61000009866C52831E
4	4	1	130	0101000020E610000050B7383BB9E
5	5	1	131	0101000020E61000000908AC4D41F
6	6	1	132	0101000020E6100000C2581F60C9C
7	7	1	133	0101000020E61000007AA99272510
8	8	1	134	0101000020E610000032FA0585D91
9	9	2	125	0101000020E6100000B623F8DE10E
10	10	2	126	0101000020E61000006E746BF198B
11	11	2	127	0101000020E610000027C5DE0321C
12	12	2	128	0101000020E6100000E0155216A9D
13	13	2	129	0101000020E61000009866C52831E
14	14	2	130	0101000020E610000050B7383BB9E

```
    FROM square_vertices_10
    ORDER BY id DESC;
```

Data Output | Messages | Notifications — Showing rows: 1 to 1000 — Page

	id [PK] integer	diagonal_id integer	vertical_id integer	geom geometry
1	14498	136	129	0101000020E61000009866C52831E140407
2	14497	136	126	0101000020E61000006E746BF198BE40407
3	14496	136	125	0101000020E6100000B623F8DE10B340407
4	14495	136	123	0101000020E6100000458211BA009C40407
5	14494	135	130	0101000020E610000050B7383BB9EC4040F
6	14493	135	129	0101000020E61000009866C52831E14040F
7	14492	135	128	0101000020E6100000E0155216A9D54040F
8	14491	135	127	0101000020E610000027C5DE0321CA4040F
9	14490	135	126	0101000020E61000006E746BF198BE4040F
10	14489	135	125	0101000020E6100000B623F8DE10B34040F
11	14488	135	124	0101000020E6100000FED284CC88A74040F
12	14487	135	123	0101000020E6100000458211BA009C4040F
13	14486	135	122	0101000020E61000008C319EA778904040F
14	14485	135	121	0101000020E6100000D4E02A95F0844040F

Додаток : справді з 1км^2 виходить кількість точок близька 100 тисяч. Через що по-перше дуже довго опрацьовується (Бо робиться циклом і можливо це треба було якось розбити частинами.) по друге html карти виходить близько 600 мб . І я просто неможу його вигрузити. Тож може все таки json і не була б така погана ідея. Також уточнення, таблиця містить виключно унікальні вершини квадратів і для покращення структури бази даних було б непогано зробити таблицю що мала б в собі співвідношеня цих точок щоб зберігати саме квадрати.

5.

Задача: Графічно відобразити ці квадрати у середовищі Python на карті або на веб-інтерфейсі за допомогою бібліотеки Leaflet.

Розбір коду**:**

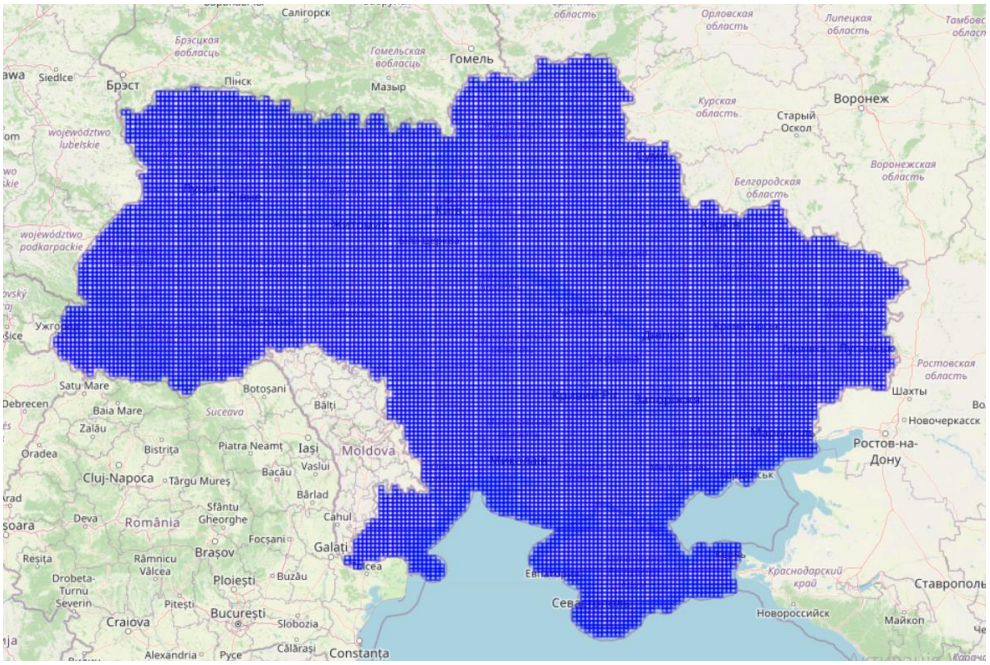
1. **Завантаження даних із бази**: Підключення до бази даних. Витягуємо координати точок квадратів із таблиці square_vertices_10. Та закриття бази
2. **Формування квадратів**: Побудова геометрії квадратів із координат точок. Для цього:
    - Організуємо точки у словник**,** щоб швидко знаходити координати за diagonal_id та vertical_id
    - Створюємо квадрати**,** знаходячи сусідні вершини.

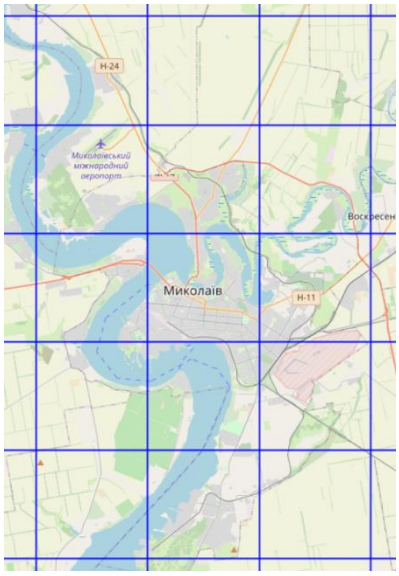
    Результат: Список squares із геометрією кожного квадрата
3. **Створення карти за допомогою Folium:** Визначаємо центр карти на основі середніх координат.Створюємо карту та додаємо полігони квадратів.
4. **Відображення карти**:Збережений файл ukraine_squares_map.html можна відкрити у браузері.Візуалізація: На карті будуть показані квадрати по ~10 км^2 , які потрапляють у межі України**.**

Консольний вивід:

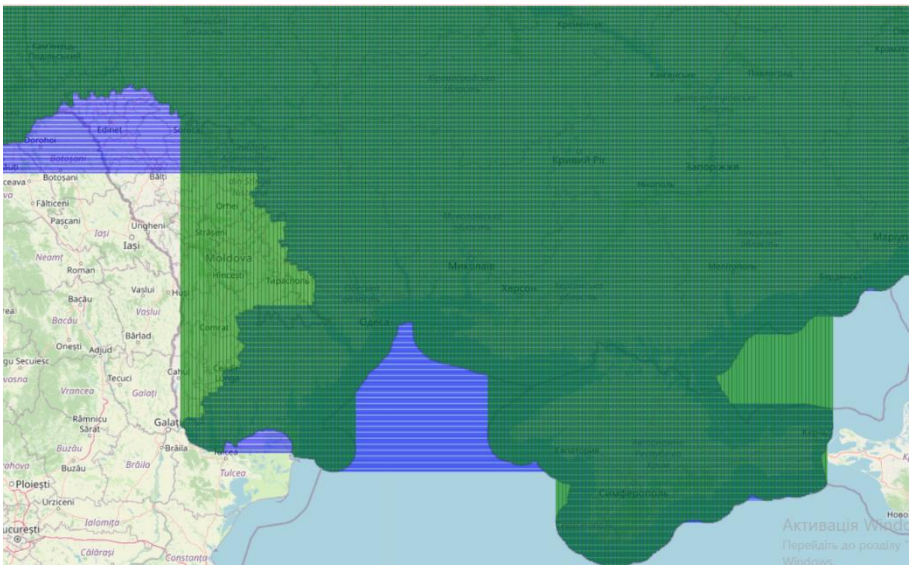
```
Карта збережена у файл 'ukraine_grid_map.html'. Відкрийте файл у браузері для перегляду.
```

HTML-карта:





Додаток: були ідеї реалізувати це малювання лініями саме а не квадратами щоб обробляти менше точок, але виникла така проблема :
*сітка на 1 км, але малювання лініями



Ідеї як виправити це не прийшло. Можна було б використати мультилінії, але тоді треба додаткові бібліатеки.
Також був варіанти просто з кожної точки малювати полігон: км вправо- км вліво ,але це не було реалізовано.

6.1.

Задача: З кожної вершини графічно зобразити 3 сектори з азимутами 0, 120 та 240 градусів, розкривши їх на 60 градусів радіусом 5 км.

Розбір коду**:**

1. **Функція для додавання секторів на карту**:Функція add_sector створює сектор у вигляді багатокутника, який додається до карти.Робота функції:

   Обчислює точки контуру сектора, використовуючи тригонометрію.

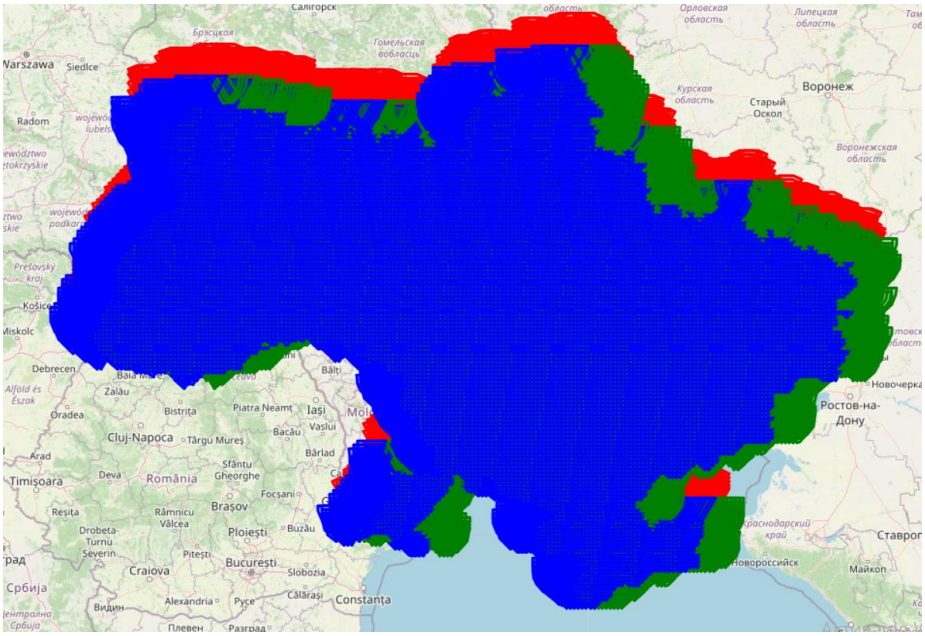
   - Для кожного кута від azimuth - spread / 2 до azimuth + spread / 2 визначає координати точки.Радіус Землі в обчисленнях прийнято за 6371 км.
   - Змінення широти (d_lat) та довготи (d_lon) залежить від кута та радіуса.

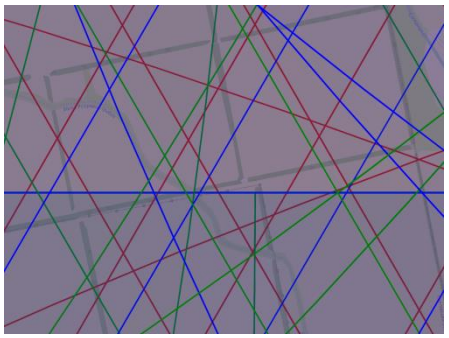
   Замкнутий багатокутник додається до карти як кольоровий полігон із напівпрозорим заповненням.
2. **Завантаження даних із бази PostgreSQL**:Підключення до бази даних coordinates_uk_db.Виконання SQL-запиту для отримання всіх точок з таблиці square_vertices_10.(Точки представлені як геометрія geom у форматі latitude (широта) та longitude (довгота)).Закриття з'єднання після отримання даних.Результатом є список intersection_points, кожен елемент якого — це кортеж із широти та довготи точки.
3. **Створення карти та додавання секторів**:Визначається початкова точка для центру карти на основі координат першої точки в списку intersection_points. Створюється об'єкт карти .Для кожної точки додаються три сектори:

   - Центральний азимут 0° (червоний сектор);
   - Центральний азимут 120° (зелений сектор);
   - Центральний азимут 240° (синій сектор).

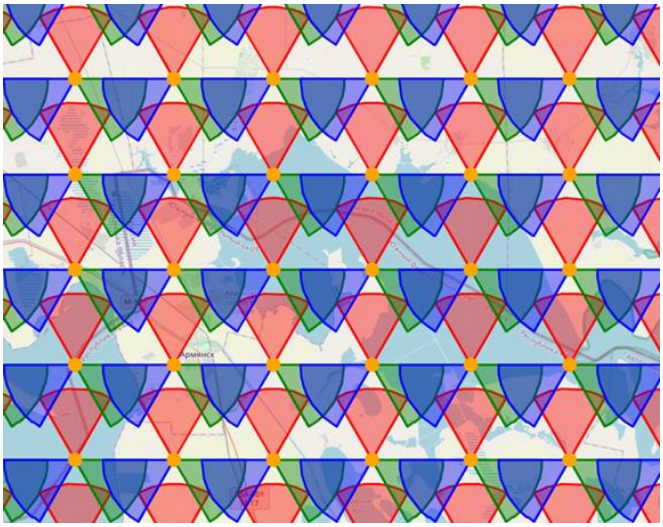
   Радіус сектора визначено як 50 км (для візуалізації), для збереження відношення 1/5.
4. **Збереження карти у файл**:Готова карта зберігається у форматі HTML в map_with_sectors.html. Її можна відкрити у браузері для перегляду.

HTML-карта:





Додаток: Маштабуємо щоб лишити відношення 1/5. Якщо буде бажання запустити на 5 км, треба замінити 50000 на 5000.



*Ось як би виглядали ці сектори з радіусом по 5 км на точках квадратів по 10 км.(також були виділені вершини помаранчевим).

6.2.

Задача:Запропонувати алгоритм, який обраховуватиме, які вершини сформованих квадратів перетинає кожен сектор. Результати перетину зберегти в БД

Розбір коду**:**

1. **Функція create_sector_polygon:** Ця функція створює WKT-репрезентацію сектора у вигляді полігону:Обчислює вершини сектора в системі географічних координат;Замикає полігон, додаючи початкову точку як кінцеву.
2. **Підготовка даних із бази PostgreSQL:** Підключення до бази. Таблиця square_vertices_10 містить усі точки.Завантажуються дані у форматі:id — ідентифікатор точки;lat, lon — координати широти та довготи.Від'єднання.
3. **Генерація секторів:**Для кожної точки створюються три сектори (азимути 0°, 120°, 240°).Кожен сектор представлено у вигляді WKT-полігону.
4. **Перевірка перетину:**Для кожного сектора перевіряється, чи потрапляють інші точки у його межі, для цього використовується функція ST_Contains бібліотеки PostGIS. Принцип перевірки:
    - ST_Contains перевіряє, чи знаходиться точка в межах геометрії сектора;
    - Умовою i.id != %s виключаються точки, які є центром сектора.
5. **Збереження результатів:**Результати записуються в таблицю sector_50. Поля:
    - id — первинний ключ.
    - sector_id — унікальний ідентифікатор сектора, де перша частина відповідає за точку з якої виходить сектор а друга за її азимут
    - point_id — ідентифікатор точки, яка перетинається сектором.

Консольний вивід:

```
Перетини секторів з вершинами обчислено та збережено.
```

Вигрузка з sector_50:

```
SELECT *
    FROM sector_50;
    --ORDER BY id DESC;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1000 Page No: 1 of

	id [PK] integer	sector_id integer	point_id integer
1	1	10	11
2	2	10	22
3	3	10	23
4	4	10	24
5	5	10	37
6	6	10	38
7	7	10	39
8	8	10	53
9	9	10	54
10	10	10	55

```
SELECT *
    FROM sector_50
    ORDER BY id DESC;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1000 Page No: 1 of 10

	id [PK] integer	sector_id integer	point_id integer
1	1088813	14498240	14492
2	1088812	14498240	14491
3	1088811	14498240	14490
4	1088810	14498240	14489
5	1088809	14498240	14488
6	1088808	14498240	14487
7	1088807	14498240	14486
8	1088806	14498240	14485
9	1088805	14498240	14477
10	1088804	14498240	14476

7.

Задача: Графічно відобразити ці сектори поверх квадратів у середовищі Python на карті або на веб-інтерфейсі за допомогою бібліотеки Leaflet.

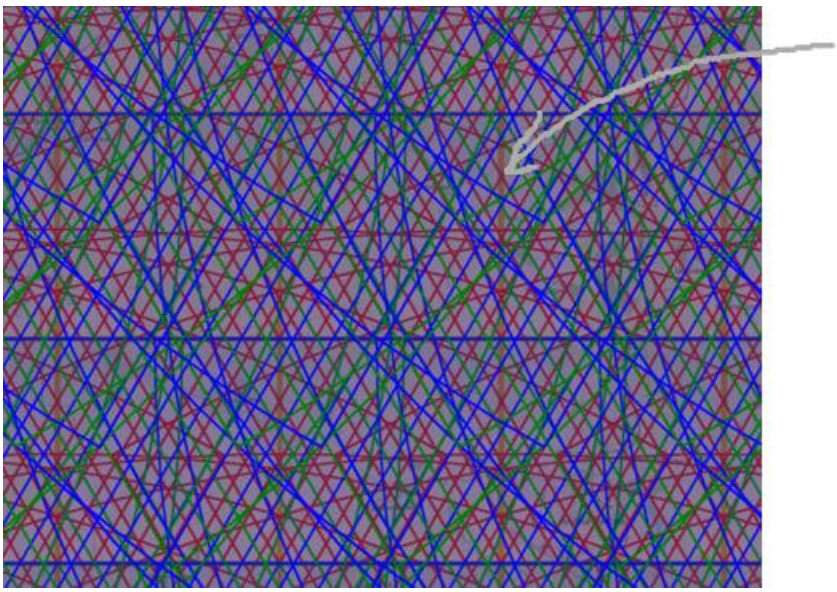
Розбір коду**:**

1. **Функція** add_sector:Додає сектор до карти.
2. **Підключення до бази даних**.: Встановлюється з'єднання з PostgreSQL через бібліотеку psycopg2.Виконується SQL-запит для отримання точок із таблиці square_vertices_10:
3. Групування точок у сітку:З отриманих даних формується сітка квадратів:
4. Визначення центру карти:Центр карти обчислюється як середнє значення широти та довготи всіх точок.
5. Малювання квадратів: Для кожного квадрата:Обираються координати чотирьох вершин;Формується список координат для замкнутого контуру квадрата.До карти додається багатокутник (folium.PolyLine), який з'єднує вершини.
6. Додавання секторів:Для кожної точки сітки:Додаються три сектори: Використовується функція add_sector.
7. 7. Створення HTML-карти:Карта зберігається у файл combined_map.html.Файл можна відкрити у браузері для перегляду.

Консольний вивід:

```
Карта збережена у файл 'combined_map.html'. Відкрийте файл у браузері для перегляду.
```

HTML-карта:



*помаранчеві полосочки від сітки кілометражу

Додаток: як на мене було б краще зробити так щоб сітка була зверху бо її під секторами ну дуже погано видно.