При виконанні цієї лабараторної роботи було використано 2 сторонні

бібліотеки numpy та matplotlib, щоб становити їх введіть у терміналі т команди:

*pip install matplotlib pip install numpy*

Перш за все у нас є основна функція *get\_convex\_hull(path),* яка приймає лише шлях звідки зчитати датасет і в кінці покаже нам результати.

У ній ми спершу зчитуємо точки з датасету за допомогою *np.loadtxt*. Потім встановлюємо розміри полотна, але оскільки за замовчуванням вони в дюймах, то є функція *pixels\_to\_inches,* яка конвертує пікселі в дюйми. Після цього я переношу ці точки на графік і створюю датасет з точками опуклої оболонки викликаючи спеціальну функцію *convex\_hull*, яка приймає всі точки і повертає лист з координатами точок опуклої оболонки.

Перейдемо до алгоритму пошуку цих точок. Я використовувала Алгоритм сканування Грема. Ось так він працює:

1. Спочатку створюємо результативний лист;
2. Тоді нам потрібно знайти початкову точку з якої ми почнемо утворювати оболонку. Очевидно, що точка, яка має якусь максимальну або мінімальну координату точно буде в ній, тому ми сортуємо наші точки і дістаємо ту в якій найменший x і відповідно y.
3. Далі сортуємо решту точок проти годинникової стрілки, що можна зробити, відсортувавши їх за нахилом (для пошуку нахилу є функція *get\_slope*.
4. Після цього розпочинається головний принцип. Ми починаємо йти по триплетам точок вони утворюють два вектори: *p1 -> p2* and *p2 -> p3* Оскільки ми рухаємось проти годинникової стрілки, ми хочемо повертати ліворуч. Поворот ліворуч означає, що цей кут багатокутника опуклої оболонки, який ми формуємо, справді є опуклим. Ми можемо дізнатися напрямок обертання, обчисливши, перехресний добуток векторів (функція *get\_cross\_product*), утворених p1, p2 і p3. Якщо результат негативний, то три точки обертаються праворуч за годинниковою стрілкою, що додало б багатокутнику увігнутий кут, тому ми хочемо позбутися другої точки, p2, оскільки вона лежить усередині опуклої оболонки. І по такому принципу ми проходимось по всіх точках залишаючи лише ті, які входять в оболонку.

Потім, отримавши, цю оболонку ми просто синім кольором з’єднуємо всі ці точки і наносимо на графік, а потім зберігаємо цей результат у файл *convex\_hull\_res.jpg* за допомогою *plt.savefig.* Оскільки в умові завдання сказано, що варто зберегти ще лише оболонку, то очищуємо фігуру і наносимо лише цю лінію, а потім за таким же принципом зберігаємо в *'just\_convex\_hull.jpg'*

Запускаючи код з датасетом, який нам даний отримуємо такі результати:



