1. Methodology

i. Language and Environment

- Python (in Spyder)

ii. Test Random Number

- N = 100
- Range of random numbers: -100 ~ 100

iii. Defined Class and Methods

- Class: ConvexHull2D
  i. reset
  ii. forward: For the purpose of removing duplicates
  iii. _quickhull: Sorting, left-most and right most points, and algorithms
  iv. _findhull: Finds the convex hull recursively
- Methods
  i. divide_area: Divide data into two set
  ii. compute_distance: Compute distance between points
  iii. triangle_partition: Partitions a set of points into two triangles based on three given points
  iv. _findhull: Finds the convex hull for a given set of points
  v. clock_sort: Sorting in clockwise order

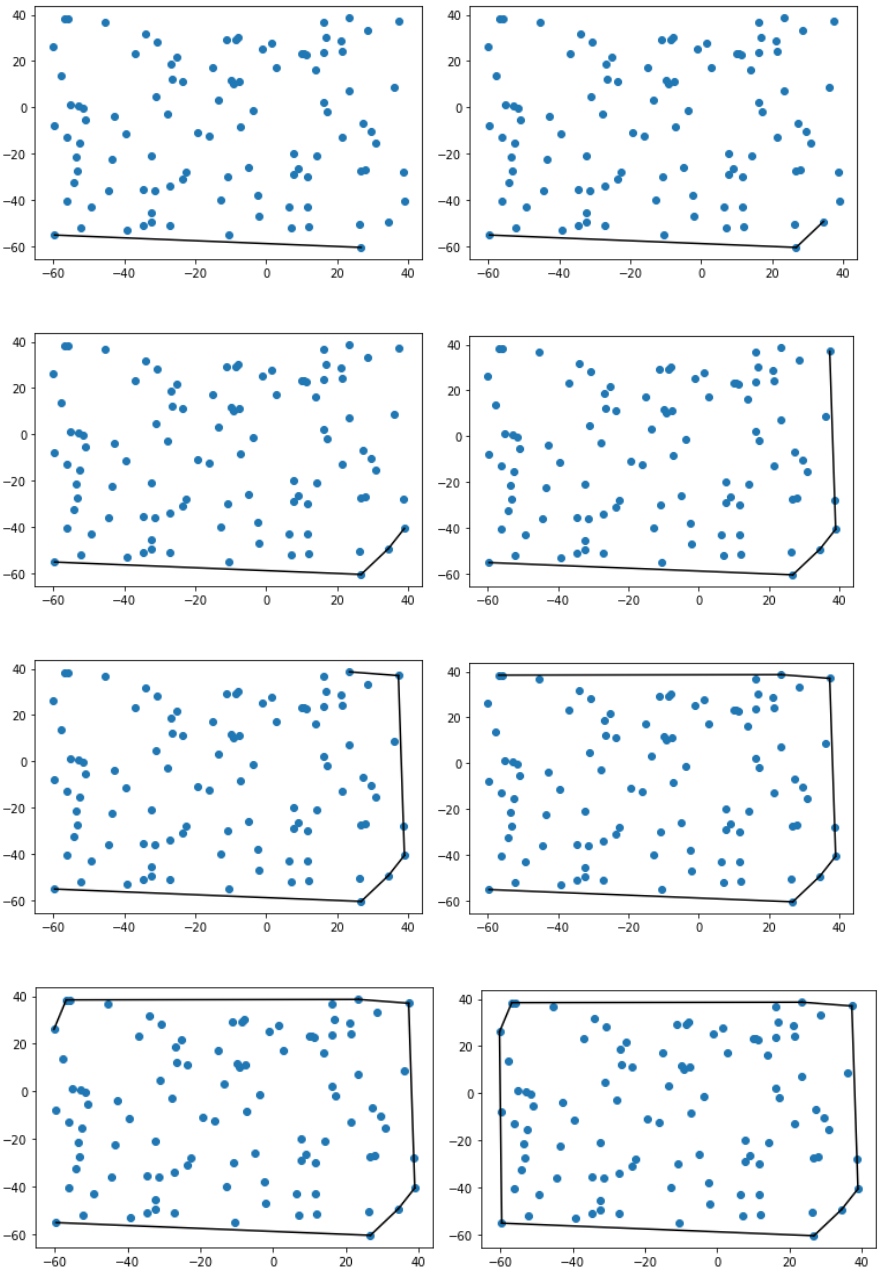
iv. Demo the Results

- Results are on the page 5.

2. Results

- Testing Data

```
The vertices of convex hull are
[[-59.65151015 -54.98604829]
 [ 26.73437132 -60.32243645]
 [ 34.44359259 -49.14308694]
 [ 38.93613833 -40.45491225]
 [ 37.23499066  37.02692707]
 [ 23.45553206  38.6314533 ]
 [-56.885889    38.43263377]
 [-60.26844165  26.08470763]
 [-59.65151015 -54.98604829]]
```



3. Challenges and Solutions

It took me a week to complete this assignment. At first, I had no idea where to start, so I searched for reference materials online. I compared the reference materials while trying to understand the logic, and finally managed to write everything without looking at the reference materials.

Looking back on the process of constructing the algorithm, the most difficult thing was dealing with the special cases that my initial code could not capture, aside from the fact that I couldn't write the code without the reference materials. For example, when testing the model, sometimes the randomly generated data points are distributed unevenly, causing the results to be unsatisfactory. This makes it difficult to draw a convex polygon. Alternatively, there may be many points collinear with one another, causing problems when connecting the lines of the convex polygon. Therefore, I was constantly thinking about how to modify the main program to capture these exceptions.

Initially, I focused on modifying the main program, but after looking at many other people's approaches, I realized that the general exceptions were simply situations where the input was None, 0, or there were less than 3 data points. These issues were different from what I was trying to solve, so I completely ignored adjusting the way the random data was generated, which could have greatly improved the problem. Later, I learned from an expert's approach: he first randomly generated a number between 0 and 1, and then enlarged it by a factor of his choosing, which approximated a normal distribution of the data. This approach can overcome many obstacles when drawing a convex polygon. In the end, I followed his approach and completed this assignment.