PES University

Cloud Computing and Big Data

Polygon Generator

Team Details

Ashwin K	- PES1UG19CS095
Ujjwal Kumar	- PES1UG19CS546
Divyansh Raina	- PES1UG19CS149
Vaibhav Raj	- PES1UG19CS554

Approaches towards the solution

I. Generating a set of random points then making a Polygon

II. Generating a set of points to make a Polygon using Turtle

Code for the solution: Github Repository with the Output screenshots and Source Code

Approach - 1

Our strategy is to make sure that the polygon includes all the randomly generated points, and that we can find an order to connect them where none of the lines intersect.

Algorithm:

1. Generate n random points
2. Find the leftmost points p
3. Find the rightmost point q
4. Partition the points into A, the set of points below pq, and B, the set of points above pq [we use the **left turn test** on (p,q) to determine if a point is above the line].
5. Sort A by x-coordinate (increasing)
6. Sort B by x-coordinate (decreasing).
7. Return the polygon defined by p, the points in A, in order, q, the points of B in order.

Total runtime is O(nlogn).

Approach - 1

Correctness:

By construction, all points besides p,q are in set A or set B.

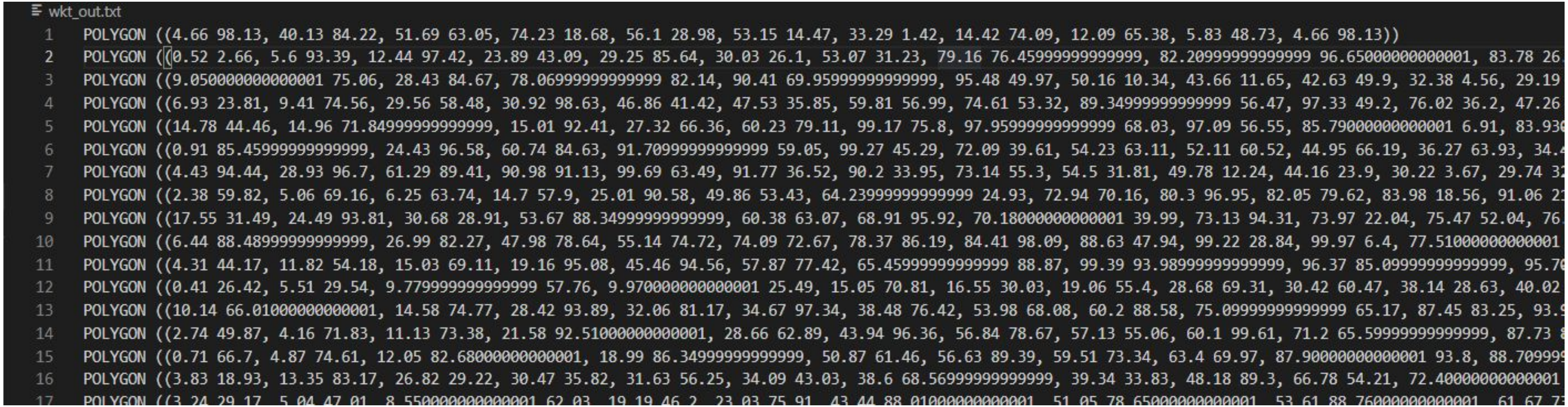
Consider each segment in the output polygon. The first edge from p to the first point in A can't intersect any segment (because there is no segment yet). As we proceed in order by x-coordinate through the points in A, from each point, the next segment is going to the right, and all previous segments are to the left. Thus, as we go from p, through all the points of A, to point q, we will have no intersections.

The same is true as we go from q back through the points of B. These segments cannot intersect each other because they proceed from right to left. These segments also cannot intersect anything in A because all points in A are below line pq, and all points in B are above this line.

Thus, no segments intersect each other and we have a simple polygon.

Output

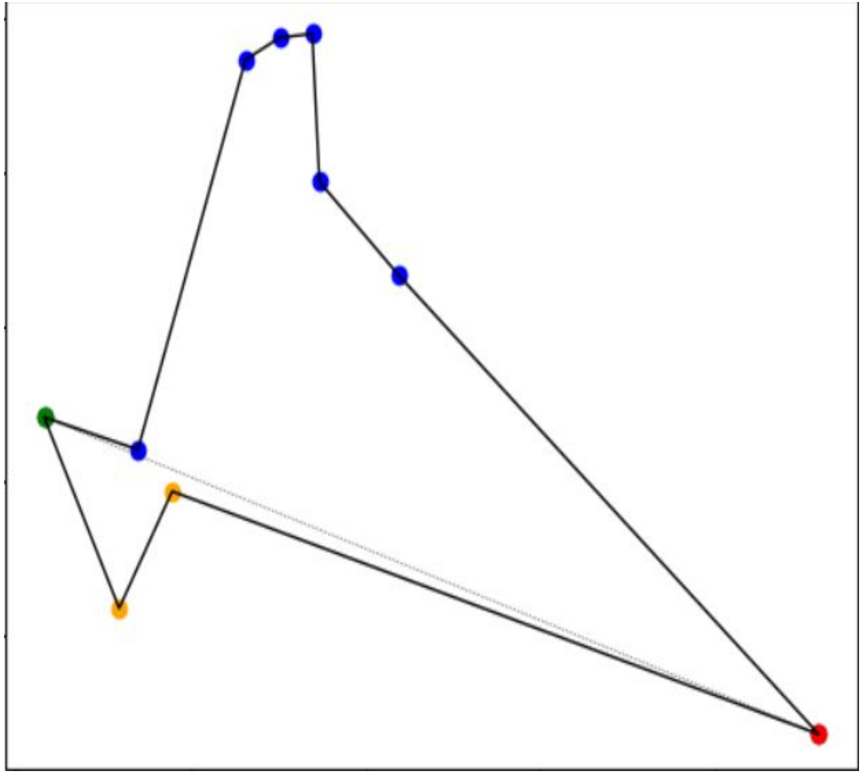
WKT text file as output

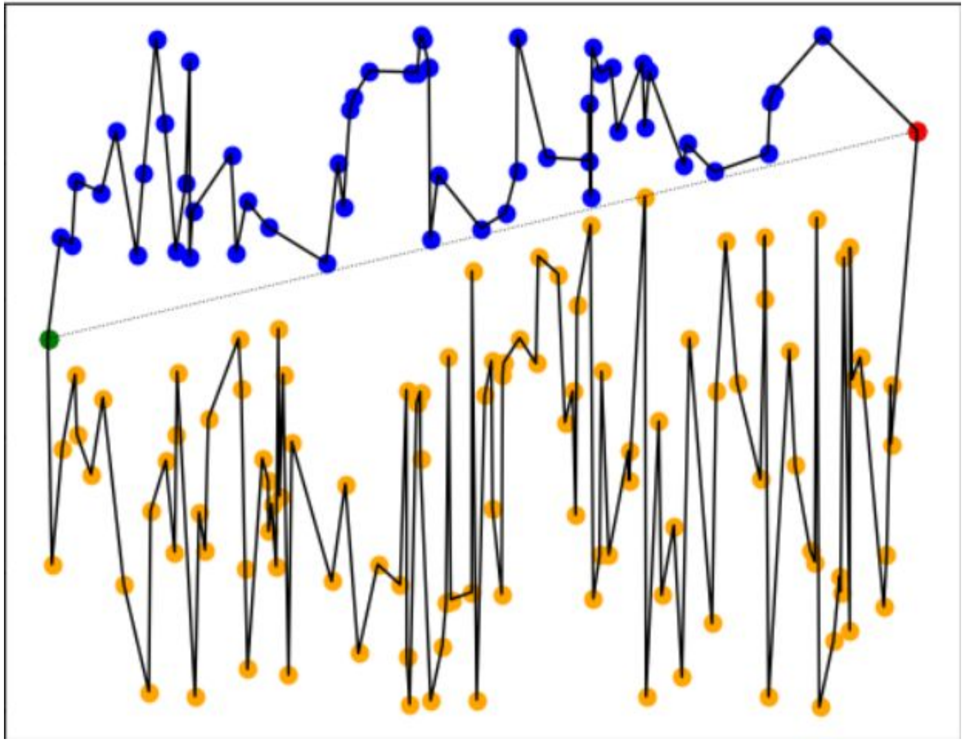


wkt_out.txt

```
POLYGON ((4.66 98.13, 40.13 84.22, 51.69 63.05, 74.23 18.68, 56.1 28.98, 53.15 14.47, 33.29 1.42, 14.42 74.09, 12.09 65.38, 5.83 48.73, 4.66 98.13))
POLYGON ((0.52 2.66, 5.6 93.39, 12.44 97.42, 23.89 43.09, 29.25 85.64, 30.03 26.1, 53.07 31.23, 79.16 76.45999999999999, 82.20999999999999 96.65000000000001, 83.78 26
POLYGON ((9.050000000000001 75.06, 28.43 84.67, 78.06999999999999 82.14, 90.41 69.95999999999999, 95.48 49.97, 50.16 10.34, 43.66 11.65, 42.63 49.9, 32.38 4.56, 29.19
POLYGON ((6.93 23.81, 9.41 74.56, 29.56 58.48, 30.92 98.63, 46.86 41.42, 47.53 35.85, 59.81 56.99, 74.61 53.32, 89.34999999999999 56.47, 97.33 49.2, 76.02 36.2, 47.26
POLYGON ((14.78 44.46, 14.96 71.84999999999999, 15.01 92.41, 27.32 66.36, 60.23 79.11, 99.17 75.8, 97.95999999999999 68.03, 97.09 56.55, 85.79000000000001 6.91, 83.93
POLYGON ((0.91 85.45999999999999, 24.43 96.58, 60.74 84.63, 91.70999999999999 59.05, 99.27 45.29, 72.09 39.61, 54.23 63.11, 52.11 60.52, 44.95 66.19, 36.27 63.93, 34.4
POLYGON ((4.43 94.44, 28.93 96.7, 61.29 89.41, 90.98 91.13, 99.69 63.49, 91.77 36.52, 90.2 33.95, 73.14 55.3, 54.5 31.81, 49.78 12.24, 44.16 23.9, 30.22 3.67, 29.74 3
POLYGON ((2.38 59.82, 5.06 69.16, 6.25 63.74, 14.7 57.9, 25.01 90.58, 49.86 53.43, 64.23999999999999 24.93, 72.94 70.16, 80.3 96.95, 82.05 79.62, 83.98 18.56, 91.06 2
POLYGON ((17.55 31.49, 24.49 93.81, 30.68 28.91, 53.67 88.34999999999999, 60.38 63.07, 68.91 95.92, 70.18000000000001 39.99, 73.13 94.31, 73.97 22.04, 75.47 52.04, 76.
POLYGON ((6.44 88.48999999999999, 26.99 82.27, 47.98 78.64, 55.14 74.72, 74.09 72.67, 78.37 86.19, 84.41 98.09, 88.63 47.94, 99.22 28.84, 99.97 6.4, 77.51000000000001
POLYGON ((4.31 44.17, 11.82 54.18, 15.03 69.11, 19.16 95.08, 45.46 94.56, 57.87 77.42, 65.45999999999999 88.87, 99.39 93.98999999999999, 96.37 85.09999999999999, 95.7
POLYGON ((0.41 26.42, 5.51 29.54, 9.779999999999999 57.76, 9.970000000000001 25.49, 15.05 70.81, 16.55 30.03, 19.06 55.4, 28.68 69.31, 30.42 60.47, 38.14 28.63, 40.02
POLYGON ((10.14 66.01000000000001, 14.58 74.77, 28.42 93.89, 32.06 81.17, 34.67 97.34, 38.48 76.42, 53.98 68.08, 60.2 88.58, 75.09999999999999 65.17, 87.45 83.25, 93.
POLYGON ((2.74 49.87, 4.16 71.83, 11.13 73.38, 21.58 92.51000000000001, 28.66 62.89, 43.94 96.36, 56.84 78.67, 57.13 55.06, 60.1 99.61, 71.2 65.59999999999999, 87.73
POLYGON ((0.71 66.7, 4.87 74.61, 12.05 82.68000000000001, 18.99 86.34999999999999, 50.87 61.46, 56.63 89.39, 59.51 73.34, 63.4 69.97, 87.90000000000001 93.8, 88.70999
POLYGON ((3.83 18.93, 13.35 83.17, 26.82 29.22, 30.47 35.82, 31.63 56.25, 34.09 43.03, 38.6 68.56999999999999, 39.34 33.83, 48.18 89.3, 66.78 54.21, 72.40000000000001
POLYGON ((3.24 29.17, 5.04 47.01, 8.550000000000001 62.03, 19.19 46.2, 23.03 75.91, 43.44 88.01000000000001, 51.05 78.65000000000001, 53.61 88.76000000000001, 61.67 7
```

Output





Output

The time taken to generate polygons of different sizes **gradually grows.**

But it is always in the range of 10^-3 to 10^-4 since the algorithm used here is of O(nlogn).

The file size for the output is around 14.1 KB.

Approach - 2

Here we use turtle in Python for our solution.

We move the pointer forward at a specific angle and construct the required polygon to check for intersection.

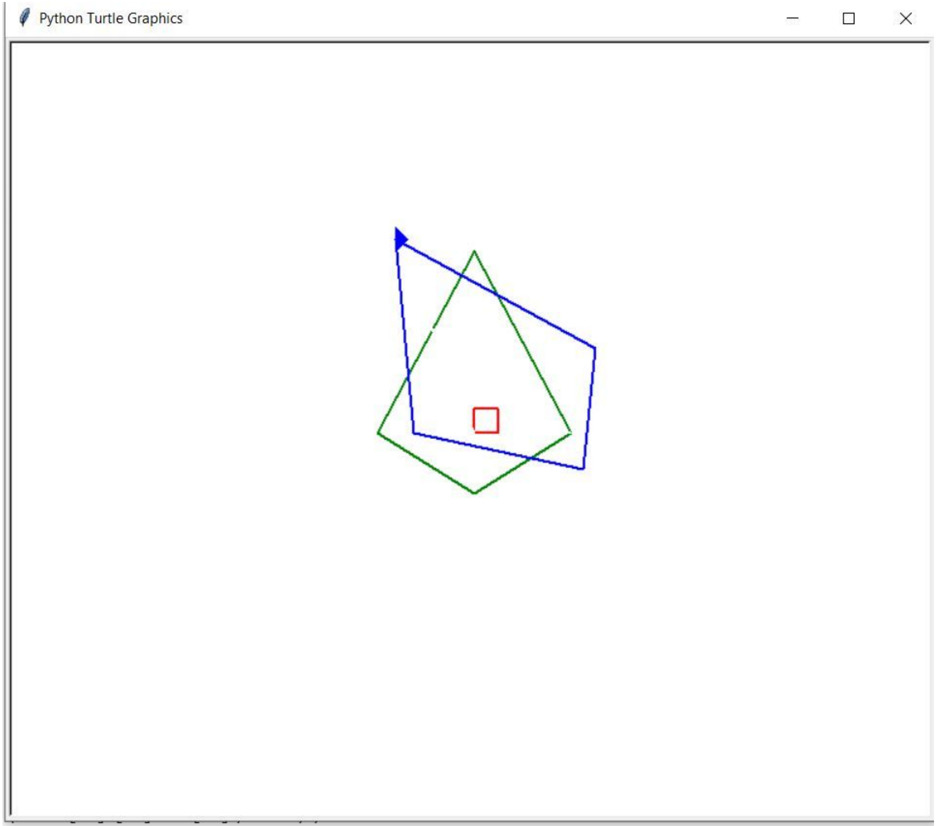
Since we are dealing with non intersecting polygons we have to check with the coordinates generated in random whether these coordinates intersect with the previous co-ordinates or not.

If they do not intersect then we go with these points or else we chose new points.

We then go to check for the quadrant with respect to the previous point and accordingly we set the quadrant

Thus we get the angle now we calculate how much should our pointer move forward ,we calculate the distance and move our pointer forward in that direction.

Output



Polygons generated using Approach 2

Conclusions

In real scenario,consider the outer polygon to be a water the inner polygon to be a small land mass.

Also calculating area can gives us insights that that inner area is less than outer area.

Based on how many points are bounded,the area and perimeter of the polygon, we can guess what might be the land mass associated with it and its neighbour.

Since we are already checking for intersecting polygons,we can provide with the information that if 2 land masses intersect then there must be a common land mass between.

If all the 4 extreme points are not bounded by one another,then there is no similarity between the land mass.

PES University

Cloud Computing and Big Data

THANK YOU!