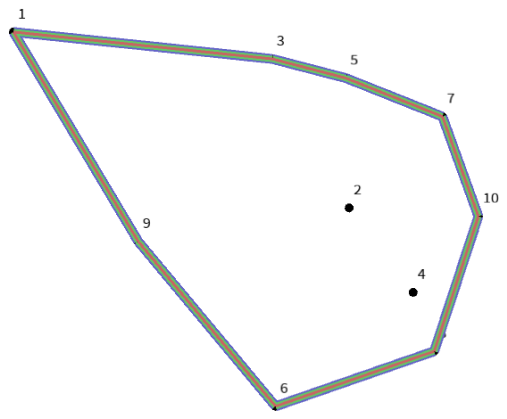
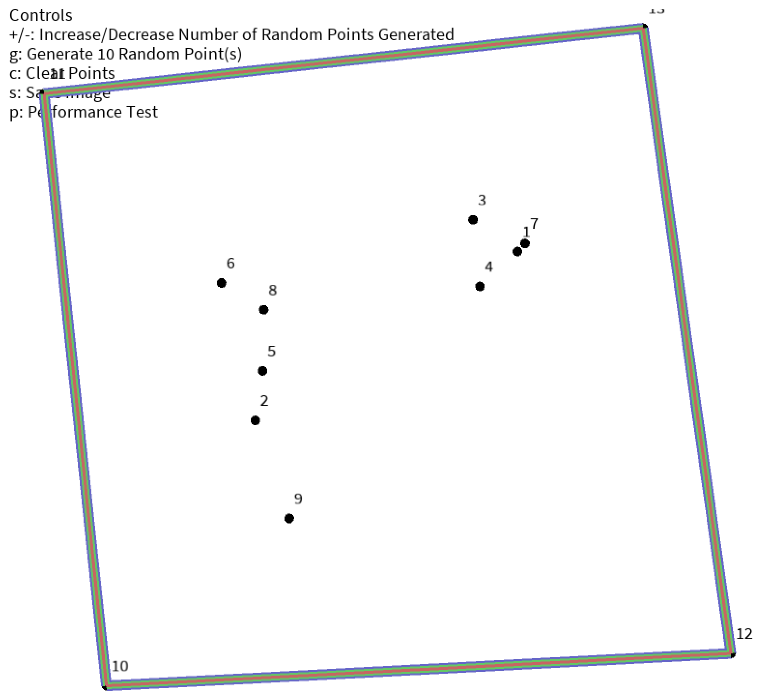
Version 1 of random points generation: random set of points evenly distributed are being generated. This helped understand the average case of the algorithms.



```
Testing with version 1 random segments:------------------------------
  10 vertices per test & 500 iterations
    QuickHull Implementation:     12 ms
    Graham Implementation:        22 ms
    Gift Wrapping Implementation: 9 ms
  100 vertices per test & 500 iterations
    QuickHull Implementation:     28 ms
    Graham Implementation:        108 ms
    Gift Wrapping Implementation: 43 ms
  1000 vertices per test & 500 iterations
    QuickHull Implementation:     177 ms
    Graham Implementation:        857 ms
    Gift Wrapping Implementation: 425 ms
```

We can see that in this usual average case graham performs the worst and for a higher number of points the quickhull performs better. Quickhull is roughly 2 times faster than gift wrapping for higher numbers and roughly 5 times faster than graham.

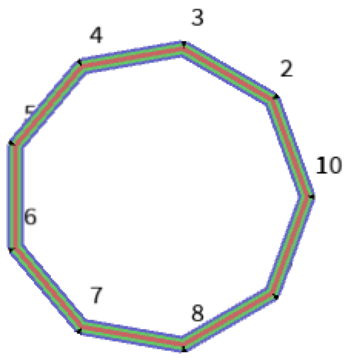
Version 2 of random points generation: the random points being generated are such that in every case the convex hull will be an exactly 4 point polygon. This helps test the performance of different algorithms as this is a specific case.


Controls
+/-: Increase/Decrease Number of Random Points Generated
g: Generate 10 Random Point(s)
c: Clear Points
s: Save Image
p: Performance Test
11
3
7
1
4
6
8
5
2
9
12
10

```
 Testing with version 2 random segments:--------------------------------
   10 vertices per test & 500 iterations
     QuickHull Implementation:     1 ms
     Graham Implementation:        5 ms
     Gift Wrapping Implementation: 2 ms
   100 vertices per test & 500 iterations
     QuickHull Implementation:     4 ms
     Graham Implementation:        67 ms
     Gift Wrapping Implementation: 20 ms
   1000 vertices per test & 500 iterations
     QuickHull Implementation:     22 ms
     Graham Implementation:        1047 ms
     Gift Wrapping Implementation: 172 ms
```

In this set of random points, quickhull always performs better and graham always performs the worst by a very significant factor. We can see that quickhull is 5 times faster than gift wrapping and almost 50 times faster than graham.

Version 3 of random points generation: the random points generated here are such that all the points will lie on the convex hull. This helps test the case where the convex hull has a max number of points.



```
 Testing with version 3 random segments:-------------------------------
   10 vertices per test & 500 iterations
     QuickHull Implementation:     3 ms
     Graham Implementation:        7 ms
     Gift Wrapping Implementation: 4 ms
   100 vertices per test & 500 iterations
     QuickHull Implementation:     68 ms
     Graham Implementation:        39 ms
     Gift Wrapping Implementation: 243 ms
   1000 vertices per test & 500 iterations
     QuickHull Implementation:     942 ms
     Graham Implementation:        290 ms
     Gift Wrapping Implementation: 14199 ms
```

We notice that for this set of random points generated the gift wrapping performs significantly worse with a higher number of points as compared to others. In this case graham performs the best as the number of points becomes significant.Graham is 50 times better than gift wrapping in this case.

Performance check of Quickhull algorithm: we notice that it performs significantly worse when there are more points on the convex hull. We can also observe that it performs the best in the case with 4 points only on the convex hull as it only has 2 iterations no matter how many points it is traversing through, which aligns with the understanding we got with theoretical analysis. We can see that more points on the convex hull makes it worse and it has to do that many iterations, which is also evident in the results shown.

```
 Testing different versions for QuickHull:-------------------------------
   10 vertices per test & 500 iterations
     Version 1:     1 ms
```

```
      Version 2:     0 ms
      Version 3:     1 ms
    100 vertices per test & 500 iterations
      Version 1:     5 ms
      Version 2:     2 ms
      Version 3:     21 ms
    1000 vertices per test & 500 iterations
      Version 1:     37 ms
      Version 2:     20 ms
      Version 3:     933 ms
```

Performance check of Graham algorithm: We see that this performs the worst when the convex hull is just 4 points as it has to compare many times in each step and performs the best when all points are on the convex hull as the comparisons get minimum.

```
  Testing different version for Graham:--------------------------------
    10 vertices per test & 500 iterations
      Version 1:     4 ms
      Version 2:     5 ms
      Version 3:     3 ms
    100 vertices per test & 500 iterations
      Version 1:     52 ms
      Version 2:     71 ms
      Version 3:     29 ms
    1000 vertices per test & 500 iterations
      Version 1:     795 ms
      Version 2:     1061 ms
      Version 3:     289 ms
```

Performance check of Gift Wrapping algorithm: We notice that it performs the worst when there are more points on the convex hull. We can also observe that it performs the best in the case with 4 points only on the convex hull as it the steps then become constant. We can see that more points on the convex hull makes it worse as the number of turns it takes to reach the starting point is maximum.

```
  Testing different version for Gift Wrapping:---------------------------
    10 vertices per test & 500 iterations
      Version 1:     2 ms
      Version 2:     3 ms
      Version 3:     3 ms
    100 vertices per test & 500 iterations
      Version 1:     32 ms
      Version 2:     19 ms
      Version 3:     157 ms
    1000 vertices per test & 500 iterations
      Version 1:     422 ms
```

Version 2:	172 ms
Version 3:	14102 ms