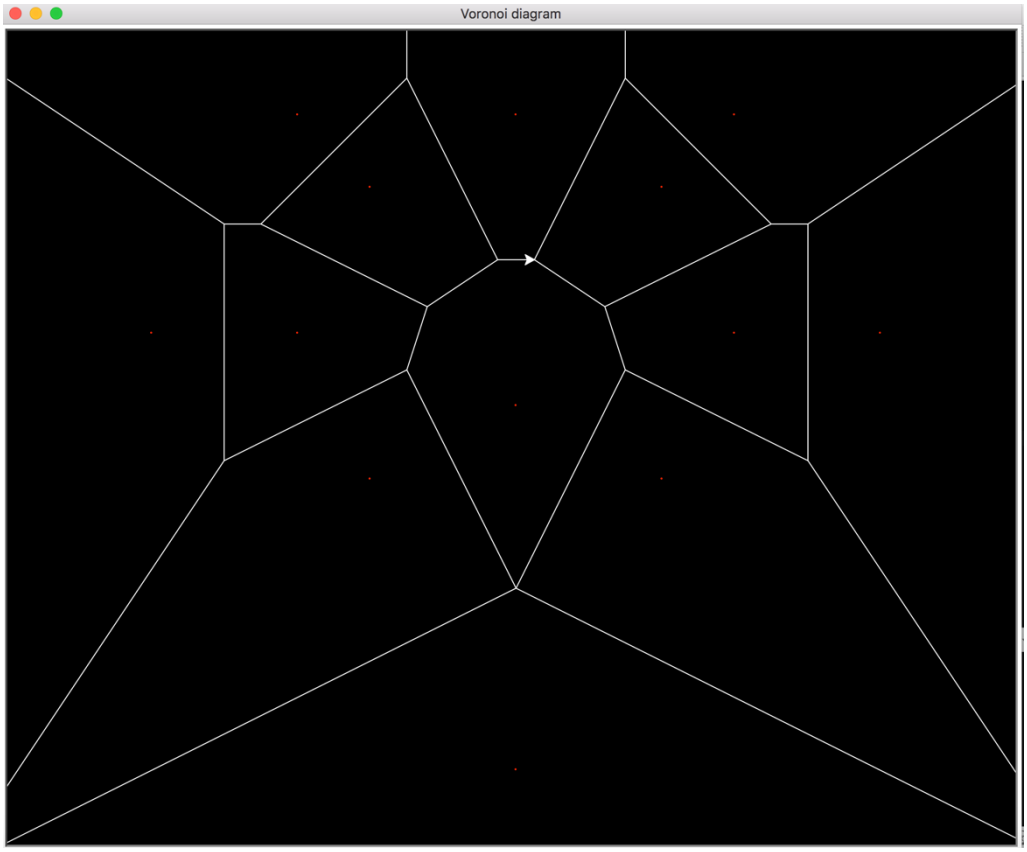
To run the file, type: python P4.py

Part 1
For the voronoi diagram part, only uncomment the drawVoronoi method and leave the main method like this.

```
#create the convex hull
ch = CreateCH(points)

#do the triangulation
Triangles = triangulate(points)

#convert to Delaunay
makeDelaunay(Triangles)

#compute the boundary of the points
bounds = computeBound(points)

#compute voronoi diagram of the delaunay triangulation (fill in the new fields in the triangle)
computeVoronoi(Triangles,bounds)

#draw voronoi diagram
drawVoronoi(Triangles,bounds,points)

#draw the shape
#drawShape(Triangles,bounds,points)

#crust algorithm
#get the union of points from delaunay triangulation vertex and voronoi diagram vertex

#crustedges = crust(Triangles,points)
#drawcrust(crustedges,points,bounds)

#output results from Triangularlist
#print("The number of Triangles is " + str(len(Triangles)))
#print("The number of vertices is " + str(len(points)))
#print("The number of edges is " + str((len(Triangles)*3+len(ch)) /2))

#visualisation
#draw(Triangles, points)
#print "end"
```

Case 1:
Sample Input:
testPoints.txt:
0 6
2 9
5 0
8 9
10 6
2 6
7 8
3 4
5 5
7 4
3 8
5 9
8 6

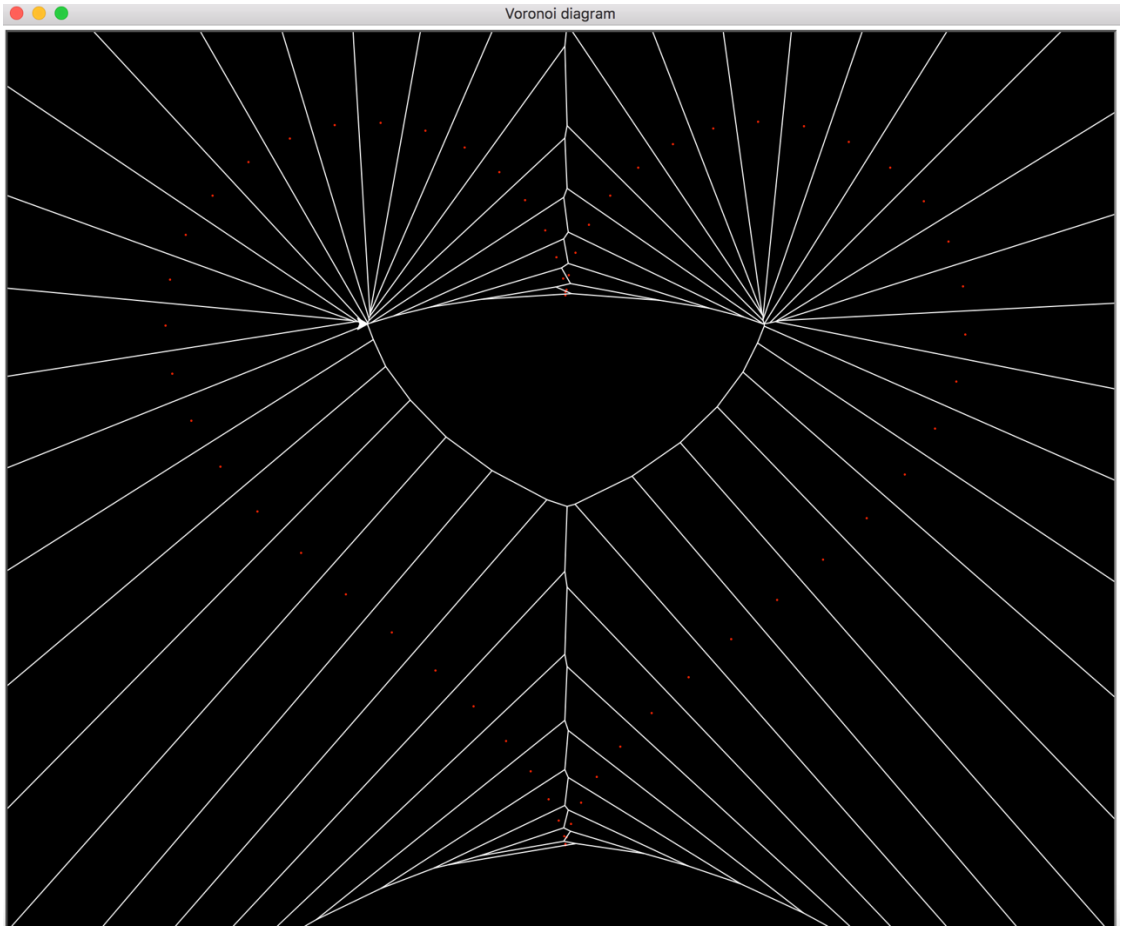
Sample Output graph:



Case 2:
Sample Input:
Points generated by the curve from mathworld
testPoints.txt:
0.0 5.0
0.0159201731166 5.20298728715
0.125462075961 5.78818260276
0.412934841426 6.68711859307
0.944863763835 7.79474338865
1.76312651683 8.98173420845
2.88031915723 10.1093721264
4.27777478026 11.0450272701
5.90642036859 11.6762670399
7.69040339459 11.9218427614
9.53317178546 11.7382927731
11.3254710973 11.1215475746
12.9545508048 10.1036372351
14.3137628617 8.74527647606
15.3116998495 7.12564032522
15.8800603099 5.33096741689
15.9795416718 3.44369720342
15.6032364547 1.53365956128
14.7772295208 -0.347572215757
13.5583432226 -2.16861122191

12.0292311147 -3.91653131029
10.2912567978 -5.59212387174
8.45579139178 -7.20322234428
6.63470348857 -8.75743222081
4.93088671122 -10.2556279925
3.42966582215 -11.687377036
2.19184308036 -13.029063734
1.24899931943 -14.244984012
0.601462169518 -15.2911473768
0.219115030239 -16.1210500702
0.0449661557514 -16.692347324
0.00115025517124 -16.97320909
-0.00318259623904 -16.9472135332
-0.0628047546177 -16.615901621
-0.266994473948 -15.9985366437
-0.690615692389 -15.1291115687
-1.38650039927 -14.0511388159
-2.37982333809 -12.8111566209
-3.66498058865 -11.4521242175
-5.20525957453 -10.0079121095
-6.93533827169 -8.49991533933
-8.76639816731 -6.93645427524
-10.5934014578 -5.31513799044
-12.3038889975 -3.62783088203
-13.7875189455 -1.86737390961
-14.9454991182 -0.0348521180406
-15.6990747214 1.85396532858
-15.9963166458 3.76432345899
-15.8166064122 5.64134042941
-15.1724186761 7.41257484055
-14.1082426566 8.99425982031
-12.6967384805 10.3005496329
-11.0324704778 11.2545865902
-9.22377498119 11.7998213689
-7.38348590508 11.9098736752
-5.61934253737 11.5953335172
-4.02493152014 10.9062673269
-2.67196649085 9.92975988962
-1.60458863108 8.78251216494
-0.836190178798 7.59922581235
-0.3490369913 6.51813150897
-0.0967156927582 5.66546481242
-0.00917818686347 5.14089084988

Sample output graph:



Part 2:

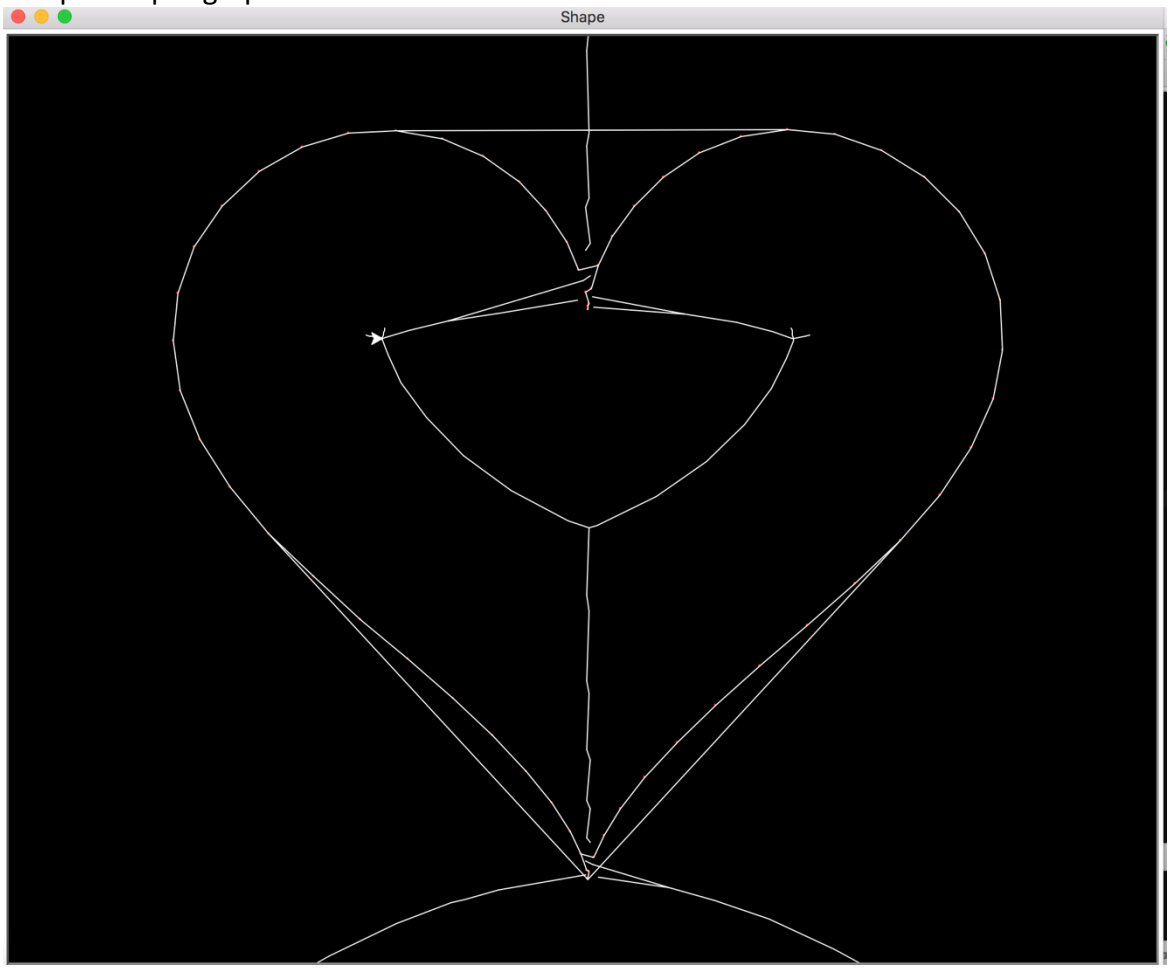
For the shape reconstruction part, only uncomment the drawShape method and leave the main method like this.

```
#create the convex hull
ch = CreateCH(points)

#do the triangulation
Triangles = triangulate(points)

#convert to Delaunay
makeDelaunay(Triangles)

#compute the boundary of the points
bounds = computeBound(points)

#compute voronoi diagram of the delaunay triangulation (fill in the new fields in the triangle)
computeVoronoi(Triangles,bounds)

#draw voronoi diagram
#drawVoronoi(Triangles,bounds,points)

#draw the shape
drawShape(Triangles,bounds,points)

#crust algorithm
#get the union of points from delaunay triangulation vertex and voronoi diagram vertex

#crustedges = crust(Triangles,points)
#drawcrust(crustedges,points,bounds)

#output results from Triangularlist
#print("The number of Triangles is " + str(len(Triangles)))
#print("The number of vertices is " + str(len(points)))
#print("The number of edges is " + str((len(Triangles)*3+len(ch)) /2))

#visualisation
#draw(Triangles, points)
#print "end"
```

Sample Input:
Points generated by the curve from mathworld
testPoints.txt:
0.0 5.0
0.0159201731166 5.20298728715
0.125462075961 5.78818260276
0.412934841426 6.68711859307
0.944863763835 7.79474338865
1.76312651683 8.98173420845
2.88031915723 10.1093721264
4.27777478026 11.0450272701
5.90642036859 11.6762670399
7.69040339459 11.9218427614
9.53317178546 11.7382927731
11.3254710973 11.1215475746
12.9545508048 10.1036372351
14.3137628617 8.74527647606
15.3116998495 7.12564032522
15.8800603099 5.33096741689
15.9795416718 3.44369720342
15.6032364547 1.53365956128
14.7772295208 -0.347572215757
13.5583432226 -2.16861122191
12.0292311147 -3.91653131029
10.2912567978 -5.59212387174
8.45579139178 -7.20322234428
6.63470348857 -8.75743222081
4.93088671122 -10.2556279925
3.42966582215 -11.687377036
2.19184308036 -13.029063734
1.24899931943 -14.244984012
0.601462169518 -15.2911473768
0.219115030239 -16.1210500702
0.0449661557514 -16.692347324
0.00115025517124 -16.97320909
-0.00318259623904 -16.9472135332
-0.0628047546177 -16.615901621
-0.266994473948 -15.9985366437
-0.690615692389 -15.1291115687
-1.38650039927 -14.0511388159
-2.37982333809 -12.8111566209
-3.66498058865 -11.4521242175
-5.20525957453 -10.0079121095
-6.93533827169 -8.49991533933
-8.76639816731 -6.93645427524
-10.5934014578 -5.31513799044
-12.3038889975 -3.62783088203

-13.7875189455 -1.86737390961
-14.9454991182 -0.0348521180406
-15.6990747214 1.85396532858
-15.9963166458 3.76432345899
-15.8166064122 5.64134042941
-15.1724186761 7.41257484055
-14.1082426566 8.99425982031
-12.6967384805 10.3005496329
-11.0324704778 11.2545865902
-9.22377498119 11.7998213689
-7.38348590508 11.9098736752
-5.61934253737 11.5953335172
-4.02493152014 10.9062673269
-2.67196649085 9.92975988962
-1.60458863108 8.78251216494
-0.836190178798 7.59922581235
-0.3490369913 6.51813150897
-0.0967156927582 5.66546481242
-0.00917818686347 5.14089084988

Sample output graph:



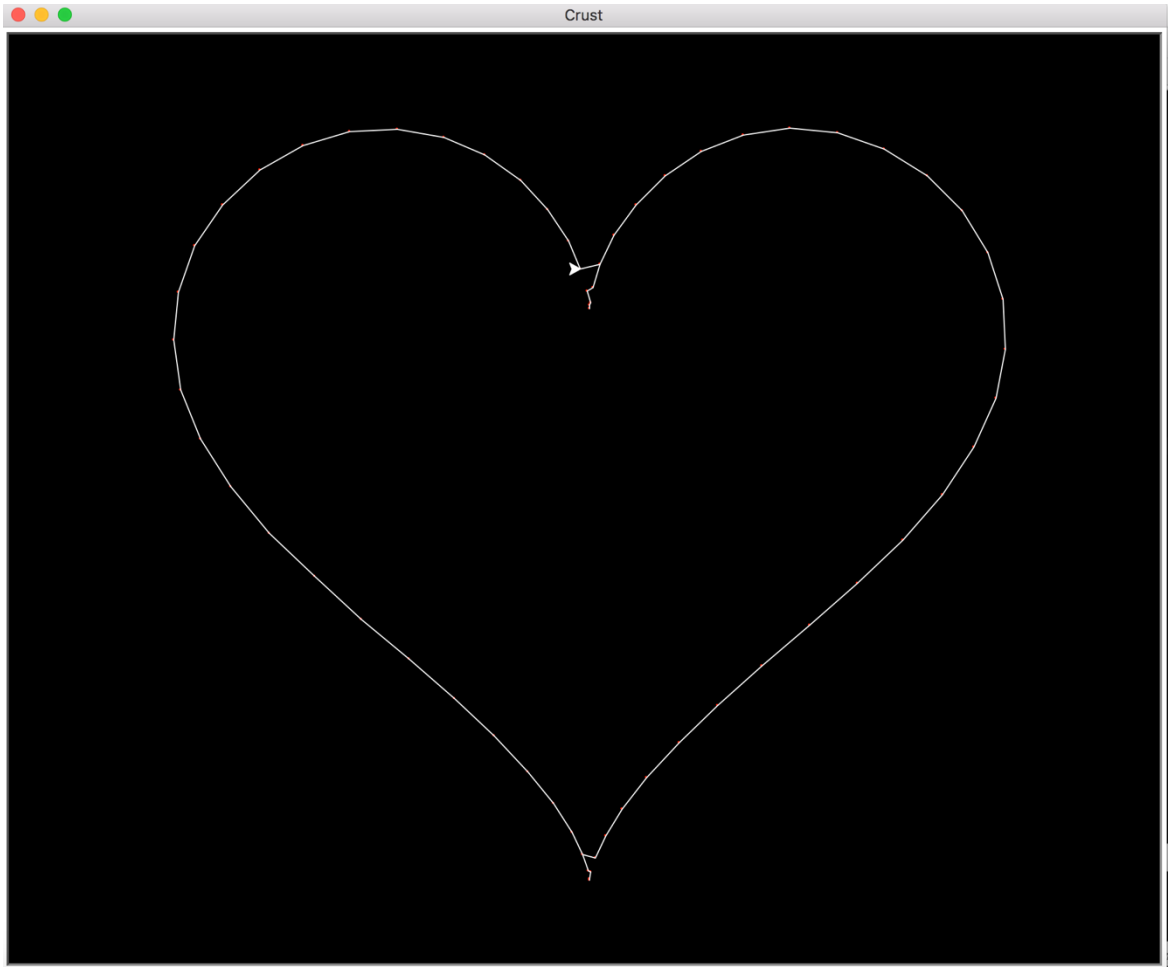
Part 3:
For the crust algorithm part, only uncomment the crust and drawcrust method, leave the main method like this.

```
#create the convex hull
ch = CreateCH(points)

#do the triangulation
Triangles = triangulate(points)

#convert to Delaunay
makeDelaunay(Triangles)

#compute the boundary of the points
bounds = computeBound(points)

#compute voronoi diagram of the delaunay triangulation (fill in the new fields in the triangle)
computeVoronoi(Triangles,bounds)

#draw voronoi diagram
#drawVoronoi(Triangles,bounds,points)

#draw the shape
#drawShape(Triangles,bounds,points)

#crust algorithm
#get the union of points from delaunay triangulation vertex and voronoi diagram vertex

crustedges = crust(Triangles,points)
drawcrust(crustedges,points,bounds)

#output results from Triangularlist
#print("The number of Triangles is " + str(len(Triangles)))
#print("The number of vertices is " + str(len(points)))
#print("The number of edges is " + str((len(Triangles)*3+len(ch)) /2))

#visualisation
#draw(Triangles, points)
#print "end"
```

Sample Input:
Points generated by the curve from mathworld
testPoints.txt:
0.0 5.0
0.0159201731166 5.20298728715
0.125462075961 5.78818260276
0.412934841426 6.68711859307
0.944863763835 7.79474338865
1.76312651683 8.98173420845
2.88031915723 10.1093721264
4.27777478026 11.0450272701
5.90642036859 11.6762670399
7.69040339459 11.9218427614
9.53317178546 11.7382927731
11.3254710973 11.1215475746
12.9545508048 10.1036372351
14.3137628617 8.74527647606
15.3116998495 7.12564032522
15.8800603099 5.33096741689
15.9795416718 3.44369720342
15.6032364547 1.53365956128
14.7772295208 -0.347572215757
13.5583432226 -2.16861122191

12.0292311147 -3.91653131029
10.2912567978 -5.59212387174
8.45579139178 -7.20322234428
6.63470348857 -8.75743222081
4.93088671122 -10.2556279925
3.42966582215 -11.687377036
2.19184308036 -13.029063734
1.24899931943 -14.244984012
0.601462169518 -15.2911473768
0.219115030239 -16.1210500702
0.0449661557514 -16.692347324
0.00115025517124 -16.97320909
-0.00318259623904 -16.9472135332
-0.0628047546177 -16.615901621
-0.266994473948 -15.9985366437
-0.690615692389 -15.1291115687
-1.38650039927 -14.0511388159
-2.37982333809 -12.8111566209
-3.66498058865 -11.4521242175
-5.20525957453 -10.0079121095
-6.93533827169 -8.49991533933
-8.76639816731 -6.93645427524
-10.5934014578 -5.31513799044
-12.3038889975 -3.62783088203
-13.7875189455 -1.86737390961
-14.9454991182 -0.0348521180406
-15.6990747214 1.85396532858
-15.9963166458 3.76432345899
-15.8166064122 5.64134042941
-15.1724186761 7.41257484055
-14.1082426566 8.99425982031
-12.6967384805 10.3005496329
-11.0324704778 11.2545865902
-9.22377498119 11.7998213689
-7.38348590508 11.9098736752
-5.61934253737 11.5953335172
-4.02493152014 10.9062673269
-2.67196649085 9.92975988962
-1.60458863108 8.78251216494
-0.836190178798 7.59922581235
-0.3490369913 6.51813150897
-0.0967156927582 5.66546481242
-0.00917818686347 5.14089084988

Sample output graph:


Crust