Nama : Andyan Yogawardhana

NIM : 21/482180/PA/21030

Kelas : KOMB1

**Tugas 11 – Convex Hull**

1. Add getConvexHullArea method in the Geometry class that takes input of the convex hull of a set of points S, CH(S), and return the area of the area enclosed by the convex hull.
2. Add getConvexHullLength method in the Geometry class that takes input of the convex hull of a set of points S, CH(S), and return the perimeter of the convex hull.

**Source Code**

1. import java.util.Arrays;
2. public class ConvexHull {
3. public static void main(String[] args) {
4. Point[] points = new Point[7];
5. points[0] = new Point(3.6, 4.5);
6. points[1] = new Point(0, 4);
7. points[2] = new Point(1.75, 6.75);
8. points[3] = new Point(2.4, 3);
9. points[4] = new Point(5.6, 5.8);
10. points[5] = new Point(0.5, 1.5);
11. points[6] = new Point(4.75, 2.1);
12. Point[] hull = Geometry.convexHull(points);
13. System.out.println("Convex Hull");
14. for (int i = 0; i < hull.length; i++) {
15. if (hull[i] != null) {
16. System.out.println(hull[i]);
17. }
18. }
19. Geometry.getConvexHullArea(hull);
20. Geometry.getConvexHullLength(hull);
21. }
22. }
23. class Point implements Comparable<Point> {
24. double x, y;
25. public Point() {
26. x = 0.0;
27. y = 0.0;
28. }
29. public Point(double \_x, double \_y) {
30. x = \_x;
31. y = \_y;
32. }
33. public int compareTo(Point other) {
34. double EPS = 1e-9;
35. double tmp;
36. if (Math.abs(x - other.x) > EPS) {
37. tmp = x - other.x;
38. if (tmp > EPS) {
39. return 1;
40. } else {
41. return -1;
42. }
43. } else if (Math.abs(y - other.y) > EPS) {
44. tmp = y - other.y;
45. if (tmp > EPS) {
46. return 1;
47. } else {
48. return -1;
49. }
50. } else {
51. return 0;
52. }
53. }
54. public String toString() {
55. return "(" + x + ", " + y + ")";
56. }
57. }
58. class Geometry {
59. public static double cross(Point O, Point A, Point B) {
60. return (A.x - O.x) \* (B.y - O.y) - (A.y - O.y) \* (B.x - O.x);
61. }
62. *// return true if pqr turns left (counter-clockwise)*
63. public static boolean ccw(Point p, Point q, Point r) {
64. return cross(p, q, r) > 0;
65. }
66. public static Point[] convexHull(Point[] P) {
67. if (P.length > 2) {
68. int n = P.length, upperLength = 0, lowerLength = 0;
69. Point[] lowerHull = new Point[n];
70. Point[] upperHull = new Point[n];
71. Arrays.sort(P);
72. *// build lower hull first*
73. lowerHull[0] = P[0];
74. lowerHull[1] = P[1];
75. lowerLength = 2;
76. for (int i = 2; i < n; i++) {
77. while (lowerLength >= 2 && !ccw(lowerHull[lowerLength - 2], lowerHull[lowerLength - 1], P[i])) {
78. lowerLength--;
79. }
80. lowerHull[lowerLength] = P[i];
81. lowerLength++;
82. }
83. *// build upper hull*
84. upperHull[0] = P[n - 1];
85. upperHull[1] = P[n - 2];
86. upperLength = 2;
87. for (int i = n - 3; i >= 0; i--) {
88. while (upperLength >= 2 && !ccw(upperHull[upperLength - 2], upperHull[upperLength - 1], P[i])) {
89. upperLength--;
90. }
91. upperHull[upperLength] = P[i];
92. upperLength++;
93. }
94. *// combine lower hull and upper hull*
95. Point[] result = new Point[2 \* n];
96. int t = 0;
97. for (int i = 0; i < lowerLength - 1; i++) {
98. result[t] = lowerHull[i];
99. t++;
100. }
101. for (int i = 0; i < upperLength - 1; i++) {
102. result[t] = upperHull[i];
103. t++;
104. }
105. result = Arrays.copyOfRange(result, 0, t);
106. return result;
107. } else if (P.length <= 2) {
108. return P.clone();
109. } else {
110. return null;
111. }
112. }
113. *// 1*
114. public static void getConvexHullArea(Point[] P) {
115. double result = 0;
116. *// deklarasi point koordinat pusat (0, 0)*
117. Point O = new Point();
118. *// menghitung luas dengan menjumlahkan cross product titik-titik yang terhubung*
119. *// (metode segitiga)*
120. for (int i = 0; i < P.length - 1; i++) {
121. result += cross(O, P[i], P[i + 1]);
122. }
123. *// untuk sisi yang menghubungkan convex point pertama dan terakhir*
124. result += cross(O, P[P.length - 1], P[0]);
125. *// hasil akhir dibagi dua agar sesuai dengan*
126. *// rumus metode luas segitiga (|a x b|) / 2*
127. System.out.println("Convex Hull Area   = " + result / 2);
128. }
129. *// 2*
130. public static void getConvexHullLength(Point[] P) {
131. double result = 0;
132. *// menghitung keliling dengan menambahkan setiap sisi convex hull*
133. for (int i = 0; i < P.length - 1; i++) {
134. result += Math.sqrt(Math.pow(P[i].x - P[i + 1].x, 2) + Math.pow(P[i].y - P[i + 1].y, 2));
135. }
136. *// untuk sisi yang menghubungkan convex point pertama dan terakhir*
137. result += Math.sqrt(Math.pow(P[0].x - P[P.length - 1].x, 2) + Math.pow(P[0].y - P[P.length - 1].y, 2));
138. System.out.println("Convex Hull Length = " + result);
139. }
140. }

Output Terminal

