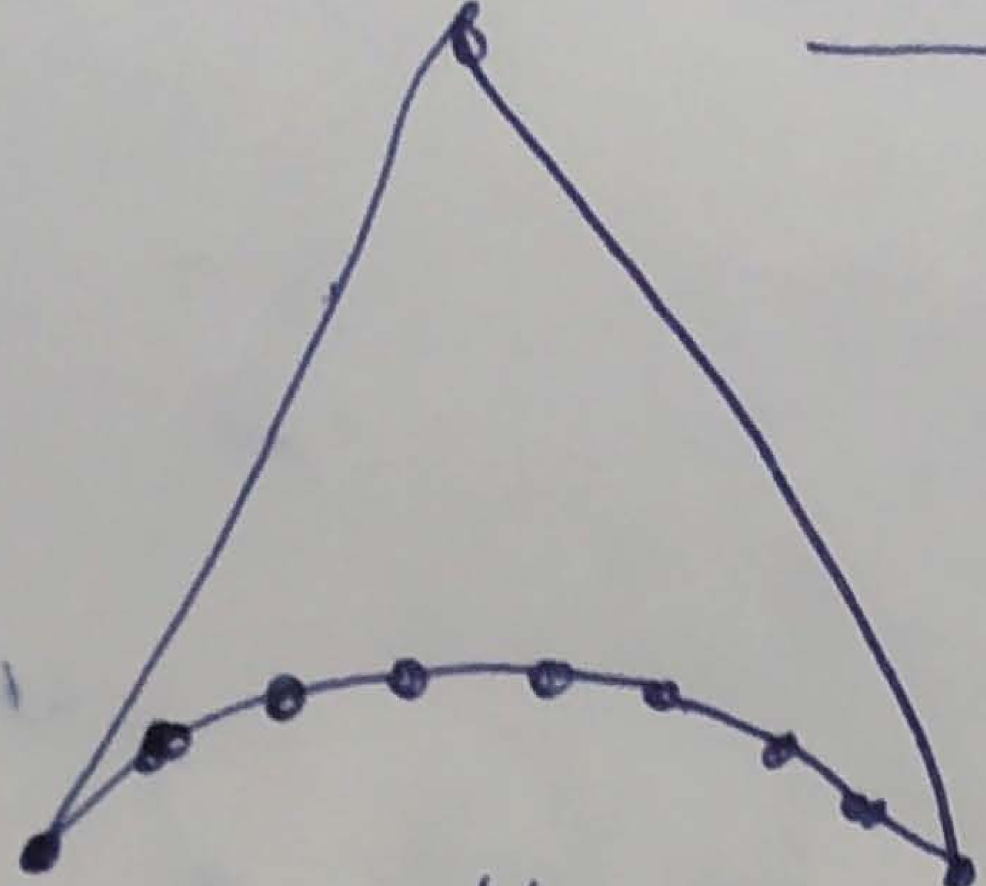
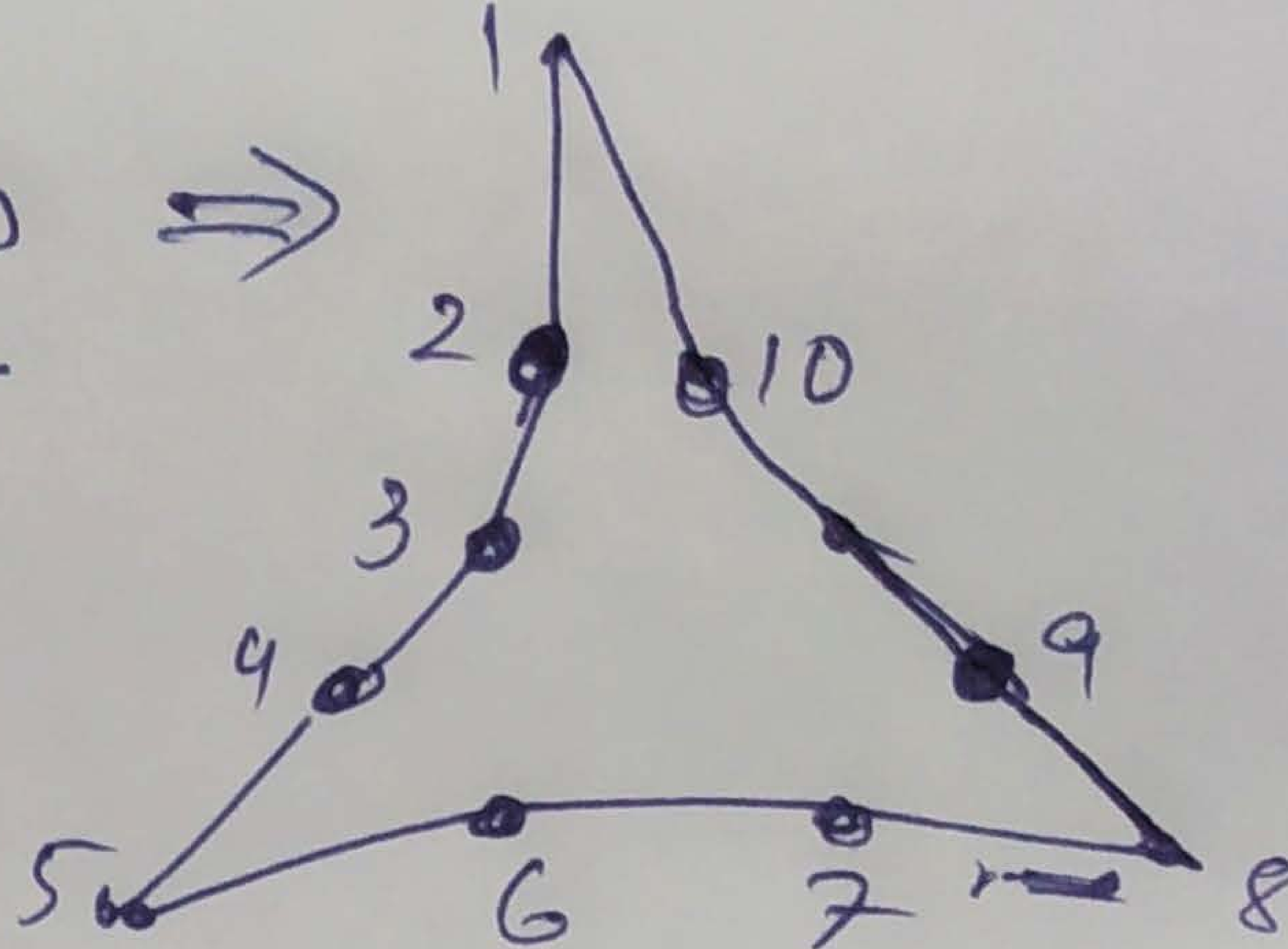


$\Leftarrow$ other solutions $\Rightarrow$



arks

reflex vertices = 7
total vertices = 10

Maximum number of reflex vertices = 7

(b) Let the # reflex vertices in P be r
convex vertices in P be c

$$r + c = 100$$

each reflex angle $\Rightarrow \frac{3\pi}{2}$; each convex angle $\Rightarrow \pi/2$

Sum of all internal angles $= r \cdot \frac{3\pi}{2} + c \cdot \frac{\pi}{2}$

arks

$$= \frac{\pi}{2}(3r + c) = \frac{\pi}{2}(3r + 100 - r)$$

$$= \frac{\pi}{2}(2r + 100) = \pi(r + 50).$$

Also from triangulation, # Δ's in P = $n-2$

Hence sum of all internal angles $= (n-2)\pi$

$$= (100-2)\pi = 98\pi.$$

Thus $\pi(r+50) = 98\pi \implies r + 50 = 98$

$$\implies \boxed{r = 48}$$

2(ii)

Algorithm for triangulation

1. Partition the interior of P as discussed before into two polygons: P_1, P_2
2. Observe that both are x-monotone polygons.
3. We can triangulate them in $O(k+n)$ time.
4. Since sorting of n vertices requires $O(n \log n)$ time, the total time needed will be $O(k + n \log n)$ time.

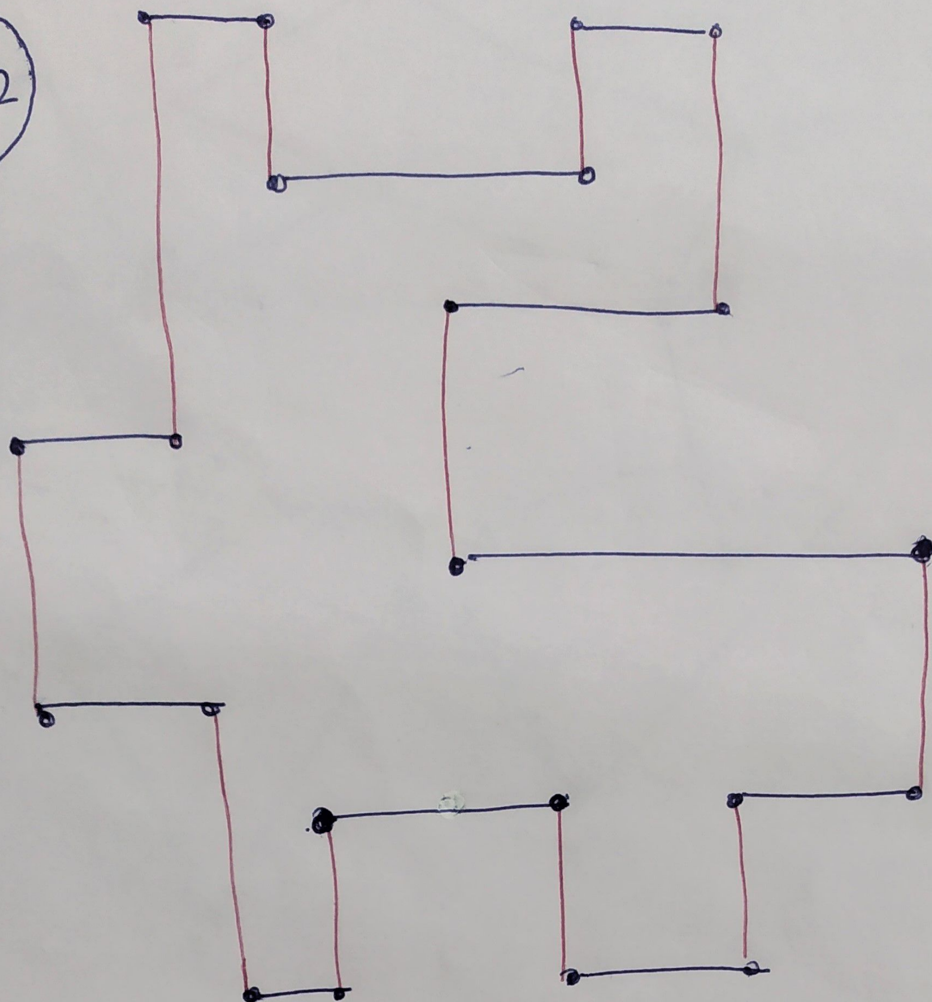
Full marks 10

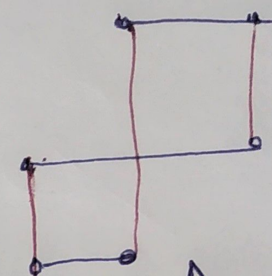
Hence, it cannot be done in $O(n + k \log k)$ time when $n > k$.

Data structure:

1) P is supplied as CCW ordering of vertices.
2) Sort n vertices using standard arrays.
3). a) For triangulating monotone polygons, maintain a ~~s~~ vertical sweep line and move it horizontally.
   (b) Merge the upper chain (lower chain) of P with the sorted list for n to get a merged sequence for nodes of P_1 and P_2 in $O(n+k)$ time.
   (c) Use a stack to triangulate monotone polygons.

Full
Marks = 12





Counterexample

1. Yes, it is constructible and unique.

2. Necessary condition: (a) # vertices along every horizontal line must be even
and (b) # vertices along every vertical line must be even.

3. Sketch of algorithm:

(i) Sort all vertices along x coordinates and along y-coordinates

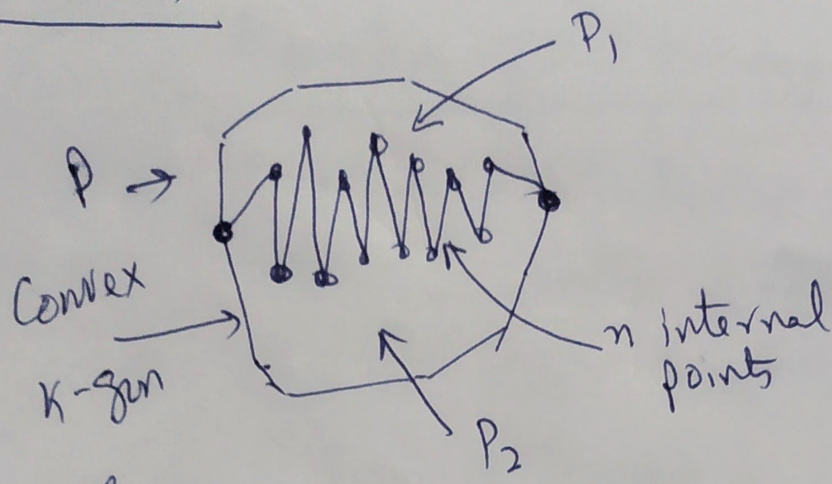
Be careful about counter examples shown above.

(ii) For each horizontal line, connect pairs starting from left.

(iii) For each vertical line connect pairs starting from top.

(iv) Sweep and connect $\Rightarrow O(n \log n)$ time.

Q. 2(i)



Solution: 1. Sort the n internal points w.r.t. x-coordinates $\rightarrow O(n\log n)$ time

2. Construct a x-monoton chain as shown.

3. Connect the leftmost vertex of the chain to a vertex of left chain of P, and rightmost vertex of the chain to a vertex of right chain of P.

Full marks 10

4. We get two polygons P_1 and P_2 as shown.

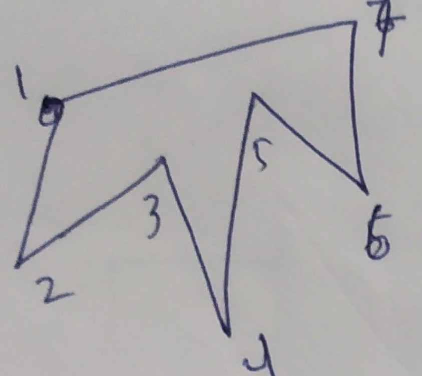
5. Triangulation of P_1 and P_2 will lead to $(3n + k - 3)$ diagonals. Hence, # edges in the triangulation will be $(3n+k-3)$ excluding the edges in the bounding polygon

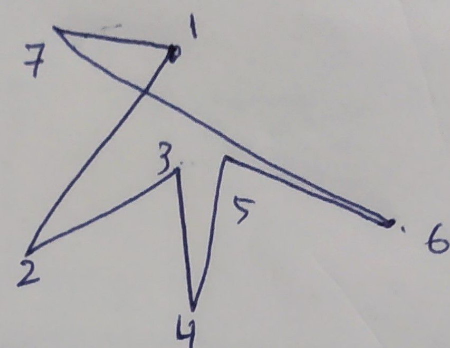
(Note: this can directly be proved using Euler's formula of planar graph also)

The number of edges in the triangulation is unique; it does not depend on the triangulation.

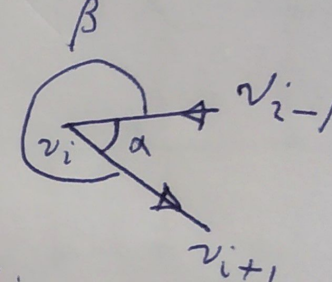
Q 3a :

Solution : Let P be given as ordered sequence of vertices.





Full Marks = 8



1. Traverse the vertices in order

2. Compute the angle α and β for three consecutive vertices v_{i-1}, v_i, v_{i+1}.

3. Compute $\sum \alpha$ and $\sum \beta$ ($\alpha \rightarrow$ left turn angles, $\beta \rightarrow$ right-turn angles. $\alpha + \beta = 2\pi$)

4. If P is a simple polygon (without any intersection, then)

$$\begin{cases} \text{either } \sum \alpha = (n-2)\pi \text{ and } \sum \beta = (n+2)\pi \\ \text{or } \sum \alpha = (n+2)\pi \text{ and } \sum \beta = (n-2)\pi . \end{cases}$$

else : P is a polygon with intersection.

This can be checked in $O(n)$ time.

Indian Institute of Technology Kharagpur

Computer Science and Engineering

CS 60064 Computational Geometry Spring 2021

Date: 30.01.2021 **Total marks** = 100

Online Test-01 Credit: 20% **Time:** 12:00 noon -1:30 PM

Instructions

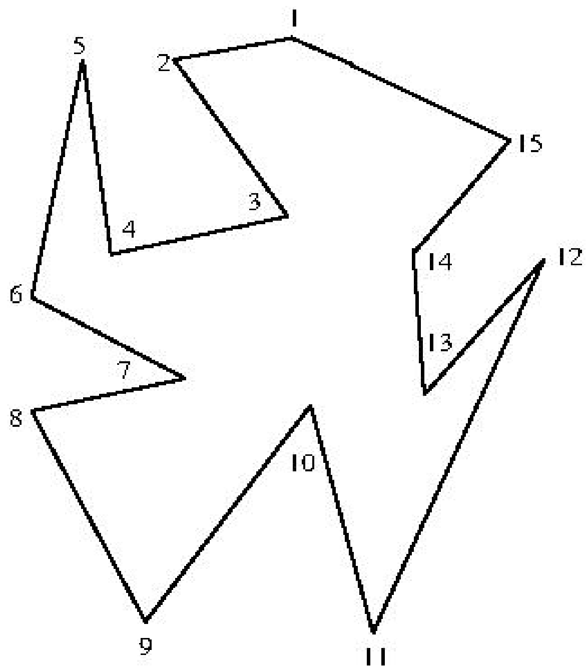
A. This is an OPEN-BOOK/OPEN-NOTES online test. Answer all questions. This question paper has three pages.

B. Submission of answers: Please create a pdf file including **your name, roll-number**, and your **answers**, and submit it to the CSE Moodle Page by **1:45 PM, Saturday, 30 January 2021.**

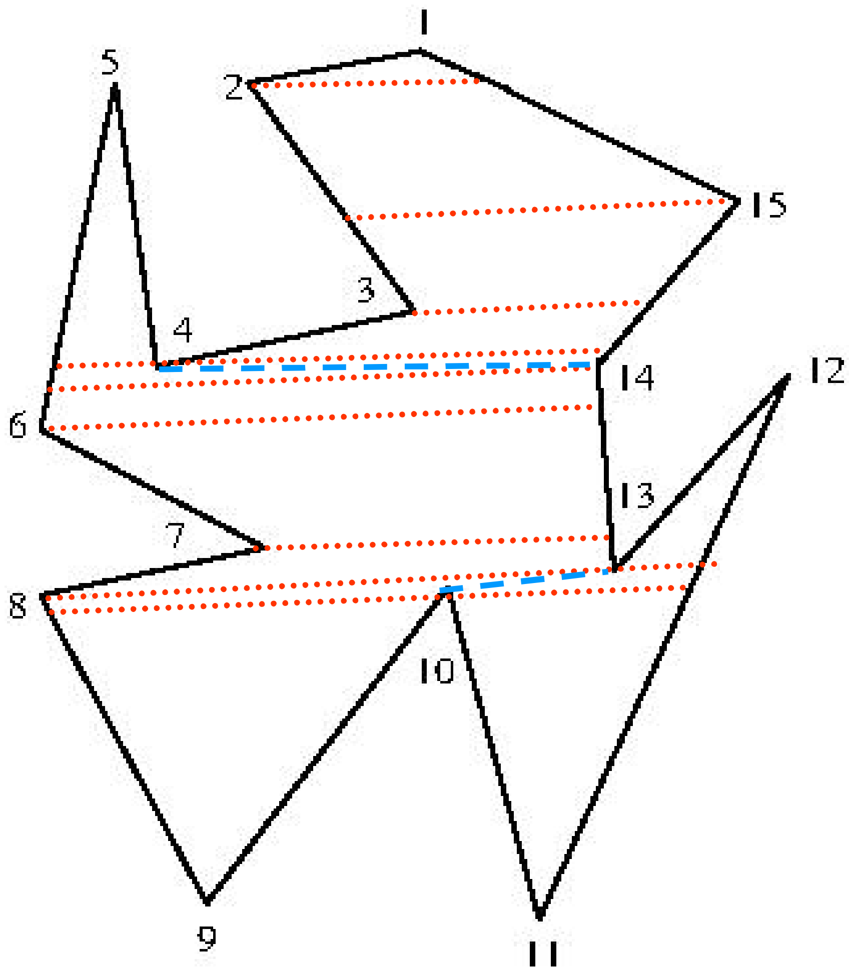
---

4. (10 points)

Consider the simple polygon *P* shown below. Show the trapezoidal decomposition and based on that, partition *P* into *y*-monotone pieces. Justify your method.



Solution to Q4:



FULL MARKS: 10

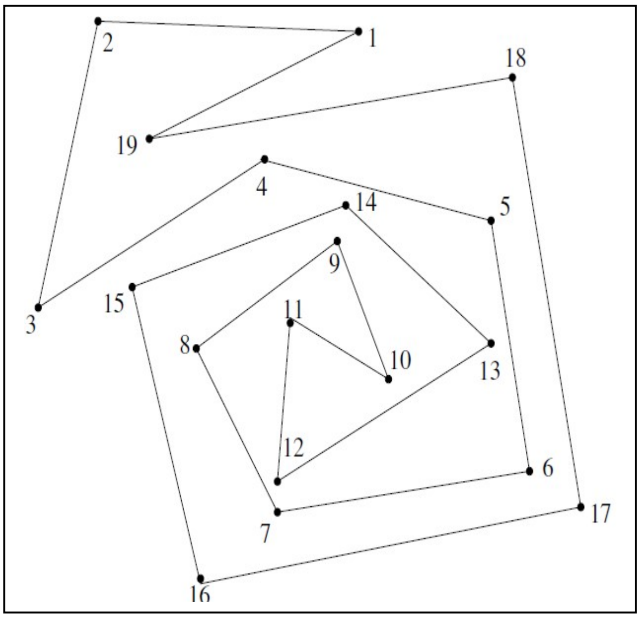
Draw horizontal line passing thru each reflex vertex and thru those convex vertices whose neighbors lie on both sides of the corresponding horizontal line.

Here reflex vertices are 3, 4, 7, 10, 13, 14; relevant convex vertices are 2, 6, 8, 15; we assume 8 and 13 share the same *y*-coordinate.

Identify reflex vertices that lies on the middle of the top bounding line of a trapezoid; here these are 4, 10, 13. For each of these find its corresponding other support vertex and join them. Hence, in this case, we put diagonals (4, 14) and (10, 13). This results in partition of the original polygon into three monotone polygons.

5. **(20 points)**

(a) Consider the simple polygon P shown below in which the vertices are numbered counter-clockwise as $\{1, 2, ..., 19\}$. Show the sequence of diagonals that are being produced when you run the triangulation algorithm based on "two-ear theorem"? How do you identify an ear? What is the time complexity of such an algorithm?



(b) Given a simple polygon P with n vertices, and a line segment λ joining its two vertices, how can you test whether or not λ is a valid diagonal?

(c) Under what condition a polygon may not have any "ear"? Justify with an example.
(10 + 4 + 6)

Solution:

(a) Three consecutive vertices $\{a, b, c\}$ will be an "ear" if ac is a diagonal.
In this example, the sequence of ears will be:

(2, 19), (3, 19), (4, 19), (4, 18), (5, 18), (6, 18), (6, 17), (7, 17), (7, 16), (8, 16), (8, 15), (8, 14), (9, 14), (9, 13), (10, 13), (10, 12); there will be 16 diagonals.

NOTE: There are other solutions as well.

Initially determine "ear-tip status" of each v_i, $O(n^2)$; Update of each ear status requires $O(1)$ ear-tip tests @ $O(n)$ per test; $n - 3$ diagonals;
Total: $O(n^2)$ time.

(b) Given a line segment λ joining two vertices of P, first check whether λ intersects with any edge of $P \Rightarrow O(n)$ time;

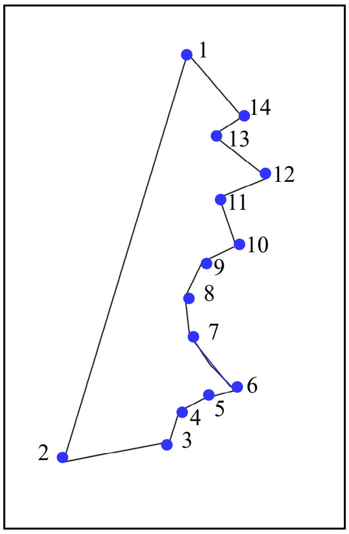
Next, check whether the mid-point of λ (or any other point on λ except the end-points) lies in the interior of the polygon $P \Rightarrow O(n)$ time; so total $\Rightarrow O(n)$ time.

(c) A simple polygon with no holes must have two ears. A *polygon with holes* may not have any "ear".

6. (20 points)
Consider the y-monotone polygon with 14 vertices shown on the right. We would like to triangulate it using a stack-based $O(n)$-time algorithm, where n denotes the number of vertices of a polygon. Assuming that the polygon is being processed by a horizontal sweep-line progressing downward from top-to-bottom,

(i) Discuss what tests/actions are being performed when the sweep-line reaches each vertex;

(ii) Write down the sequence of diagonals that will be added until the triangulation is completed. For example, the first diagonal that will be added is (13, 1). (10 + 10)



(i) Monotone polygons can be triangulated in $O(n)$ time using a stack.
Method:

- Sort vertices by increasing y-coordinates (merging the left and right chains in $O(n)$ time);
- Perform plane-sweep from top to down, halting at vertices in order;
- Triangulate everything by adding valid diagonals to the top of the sweep line; maintain a stack and use orientation tests;
- Store in a **stack** (sweep line status) the vertices that have been encountered but may need more diagonals; Un-triangulated region has a **funnel shape**; The funnel comprises a line segment (one side) and a **reflex chain** (interior angles >180°) on the other side, which is pushed into the stack;
- **Update:** *Case* 1*:* new vertex lies on chain opposite of reflex chain; triangulate; pop-off occluded vertices;
- **Update,** *Case* 2**:** new vertex lies on reflex chain
  - **Case *a*:** The new vertex lies above line through previous two vertices; triangulate as far as possible (via orientation tests); pop-off occluded vertices
  - **Case *b*:** The new vertex lies below line through previous two vertices; add to the reflex chain (stack);

(ii) There will be (14 – 3) = 11 diagonals in this case. The sequence of diagonals will be:
(13, 1), (11, 13), (11, 1), (9, 11), (9, 1), (8, 1), (5, 7), (4, 7), (4, 8), (4, 1), (3, 1);
Note: There may be some confusion regarding visibility (depending on how the vertices are drawn) and consequently, the sequence may vary.