On the structure of this practise, and on how to submit the solutions

This practise consists on exercises identified by number. You need to upload in canvas **one single `.zip` file** with the filename

`{your-full-name}-graded-practise-2.zip`.

This zip file should contain **only** directories named `exercise01`, `exercise02`, ..., `exercise10`, etc. The names of the directories **have to be** the lower case word "exercise" followed (without spaces) by a 2-digits number indicating the number of exercise.

Inside each directory, please write the code and/or the explanations required by the exercise. When an explanation is required, **please use PDF FORMAT**.

Each exercise has a *number of points*. While the perfect score is 100 points, you can try to score *more* than 100 by implementing the *bonus* problems. The BONUS problems are optional, you can *exceed* the perfect score by implementing them. In order to pass the practise, there is a *minimum* score mandatory in the non-optional problems. Independently on the total score, the minimum required in the non-optional problems is 70 points.

Aspects that will be evaluated on each solution

- Correctness.
- Clarity in the code and/or in the explanations. Easiness to read and understand the code.
- When relevant: test-cases generated to confirm the correctness of the algorithm implemented.

(100 points) Non-optional problems. [1]

1. (5 points) Write a function that receives 4 points (given by their coordinates) $a = (x_a, y_a)$, $b = (x_b, y_b)$, $c = (x_c, y_c)$, $d = (x_d, y_d)$, and returns TRUE if and only $\overline{ab} \cap \overline{cd} \neq \emptyset$.
2. (5 points) Write a function that receives a list of vertices of a polygon, given by its coordinates, and decides if the polygon convex or not.
3. (5 points) Given a line L, specified by one point $L_1 \in P$ and one direction $d = (d_x, d_y, d_z)$, compute the distance from a point $P = (P_x, P_y, P_z)$ to L.

[1] A score of at least 70 points in the non-optional problems is required in order for the current practise to be considered as passed, independently on the total score obtained, which consider also the score obtained in the optional problems.

4. (5 points) Given a triangle T (given by its vertices) and an input point P, decide if $P \in T$.

5. (5 points) Given a non-crossed polygonal region P and a points A, write a function to decide if $A \in P$.

6. (20 points) Implement the Jarvis march and the Graham scan, with and without integer points elimination.

   Measure the runtime of each algorithm with the same inputs, containing 1k, 10k, 100k, 1M, 2M, 5M points. Consider different disposition for the points in the inputs:

   - random points in a circle
   - random points in the border of a circle
   - random points in a rectangle
   - random points in the border of a rectangle
   - random points inside a region limited by a parabola
   - random points on a parabola.

   Make a table comparing the runtimes.

7. (10 points) implement a function that decides if there are intersections in a set of line segments, with runtime proportional to $N \log N$ (where N is the number of segments in the set)

8. (10 points) Implement a function that compute the 2 closest points in a given set of N points, with $1 \leq N \leq 1000000$.

9. (10 points) Explain what are the Voronoi diagram and the Delaunay triangulation. Explain how can the Delaunay triangulation be computed, and how can it be used in order to compute the Voronoi diagram.

10. (5 points) Write a function that computes the number of triangulations of a n-regular polygon. Explain your solution.

11. (5 points) Write a function that computes the area of a convex polygon.

12. (5 points) Write a function that computes the area of the intersection of two convex polygons.

13. (5 points) Implement Douglas-Peucker

14. (5 points) Implement Visvalingam-Whyatt

(40 points) Optional (bonus) problems

15. (5 points) Given a set S of segments given by the coordinates of their endpoints, and given two point A and B, decide if it is possible to go from A to B without crossing any of the segments in S.

16. (5 points) Given a set S of segments, compute the maximum number of segments in S than can be intersected by one single line.

17. (5 points) Given a set S of lines, compute in how many regions the lines of S divide the plane.

18. (5 points) What is the maximum number of different regions that we can have in the plane, after drawing N lines?

19. (10 points) Implement an algorithm that computes (in less than 1 second) the area covered by N rectangles (all of them with sides parallel to the axis), for $1 \leq N \leq 10^9$.

20. (10 points) Implement the beach-line-based approach for computing the Voronoi diagram.