# Algorithm Design and Analysis (CS60007): 2022-2023 Autumn
# Assignment 2

Submission deadline: 10-Sep-2022, 11:55 PM

Your name (Roll number: )

## Convex hull

In this assignment, you have to write a single code in C / C++ / Python for the following problems. It has to run on a given `.txt` file as input and save the output files in SVG format, as mentioned below.

Three input files are provided for this assignment. You have to run and produce outputs for all of them and submit the code plus all `.svg` files (in total, $1 + 3 \times 3 = 10$ files) as a single zip file with the name `<your roll number>-a2.zip`. For example, if your roll number is `22CS0123`, then the zip filename should be `22CS0123-a2.zip`.

Your code should be modular and friendly for evaluation. In particular, for every part, your code must have an exclusive function with proper commenting.

### Part 1 $\boxed{5\ \textbf{marks}}$

The input is a set $S$ having $n$ points in 2D, supplied as a plain text file in the following format:
1st line contains a positive integer, $g$.
2nd line it contains a positive integer, $n$.
After that, each line contains the $(x, y)$ coordinates of a point, which are both integers and divisible by $g$. So, the file contains $n + 2$ lines in total; see, for example, the file `aero.txt`.

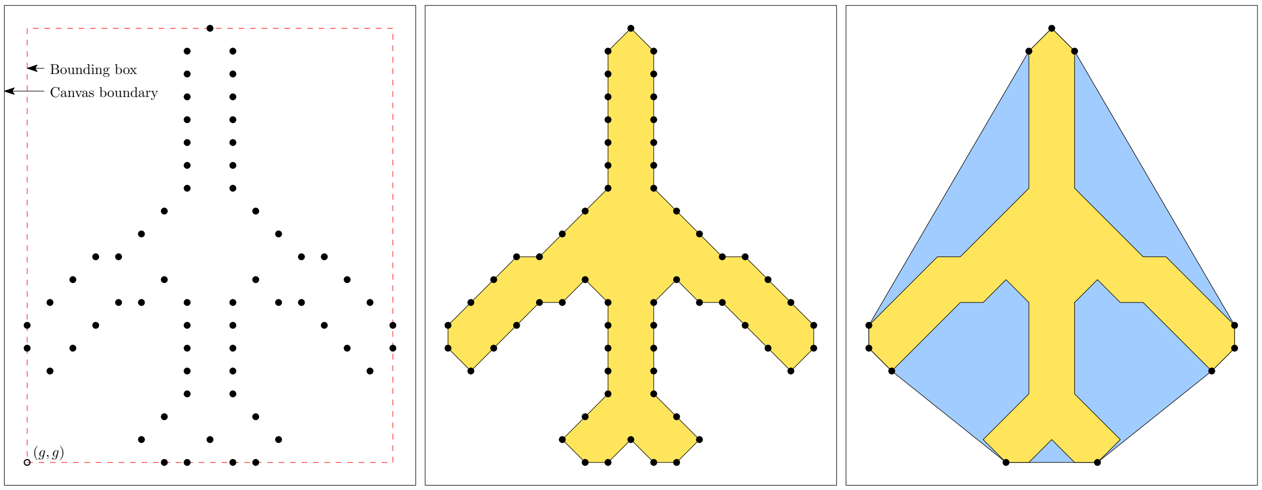
The point set $S$ is such that for any point $p \in S$, there are exactly two points in $S$ having distance $g$ or $\sqrt{2}\,g$ from $p$. Those two points should be the adjacent vertices of $p$ when we construct a simple polygon $P$ with vertex set as $S$, as described in Part 2.

Your code has to first translate $S$ so that the bottom-left corner of its bounding box is at $(g, g)$ after the translation. It may be the top-left corner depending on the coordinate system in SVG format, which you can check and fix accordingly in your code. Its canvas should have a margin of $g$ pixels—all around the bounding box of $S$—as shown in Fig. 1.
Example: If the respective width and height of the canvas are 840 and 640, then it may be declared as follows:

```
<svg width="840px" height="640px" xmlns="http://www.w3.org/2000/svg" version="1.1" >
```

Henceforth, $S$ denotes the set after translation.

Your code should save the set $S$ as an SVG file with the name `<input filename>_1.svg`. For example, if the input filename is `aero.txt`, then it should be named `aero_1.svg`.



Point set $S$ (`aero_1.svg`) Simple polygon $P$ (`aero_2.svg`) Convex hull of $P$ (`aero_3.svg`)

Figure 1: Result on `aero.txt`.

## Part 2 $\boxed{\text{5 marks}}$

This part of your code has to construct a simple polygon $P$ whose vertex set is $S$. A *simple polygon* means its edges can intersect each other only at its vertices. Note that the points in $S$ may be arbitrarily ordered, i.e., they may not appear in $S$ as the sequence of vertices of $P$.

The polygon $P$ should be colored and stored as an SVG file with the name `<input filename>_2.svg`. The vertices and the edges of $P$ should be displayed in black. As $S$ is the vertex set of $P$, there may be three or more consecutive vertices in $P$ that are collinear.

**Note** This is doable in $O(n \log n)$ time (can you think, how?) but it's okay if your code runs in $O(n^2)$ time.

## Part 3 $\boxed{\text{10 marks}}$

Using the divide-and-conquer algorithm, compute the convex hull CH($P$) of $P$. In the SVG file produced by your code, the edges of both $P$ and CH($P$) should be shown along with the vertices of CH($P$), and all other vertices of $P$ should be hidden. CH($P$) should be colored differently than $P$ and should lie below $P$ in the SVG for a proper visualization. The SVG file should be named as `<input filename>_3.svg`.

**Note** For a polygon, the convex hull can be computed in $O(n)$ time, but as the divide-and-conquer algorithm will consider the vertices of $P$ as an ordinary point set, your code will take $O(n \log n)$ time, which is acceptable.

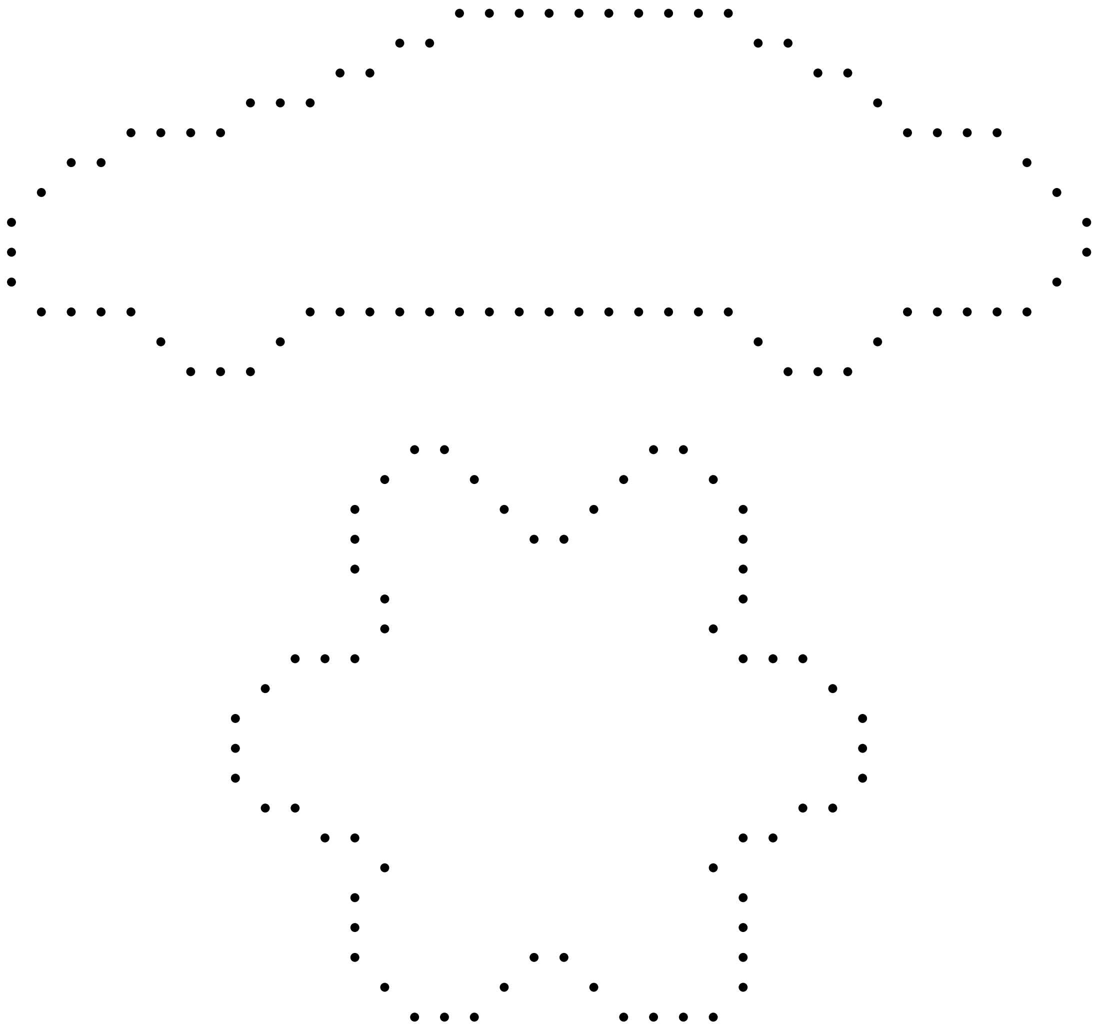


Figure 2: Input files `car.txt` and `flower.txt`.