Übungsblatt 4

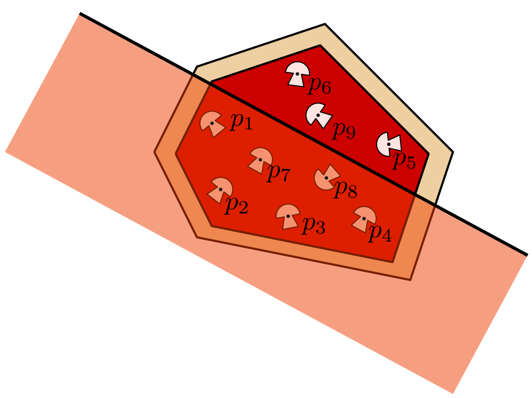
Abgabe bis 21. Dezember 2023

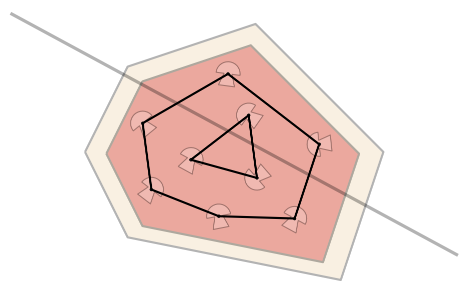
Aufgabe 1: Pizza-Guillotine Teil 2 — $5 + 5 = 10$ Punkte

Die Pizza-Guillotine hat nun den Betrieb aufgenommen und die jüngste algorithmische Verbesserung wurde in den Betriebsablauf eingebaut: nach linearer Vorberechnungszeit (Eckpunkte der Pizza in Array speichern) kann mittels einer angepassten binären Suche nach jeder zufälligen Neuausrichtung der Klinge in logarithmischer Zeit ermittelt werden, wo die Klinge die Pizza (ein konvexes Polygon) trifft.

Teilaufgabe (a) Um mehr Zeit zu sparen, sollen nun Ofen und Klinge kombiniert werden, sodass die Pizza während des Schnitts auch gleich gebacken wird! Aus Kostengründen kann allerdings nur eine Seite der Klinge mit der Ofenfunktion ausgestattet werden.

Damit die n Zutaten auf der Pizza, deren Positionen durch die Punkte $p_1, \dots, p_n$ definiert sind, der Hitze nicht mehrfach ausgesetzt werden, muss bei jedem möglichen Schnitt festgestellt werden, welche k Zutaten auf der Ofen-Seite der Klinge liegen.



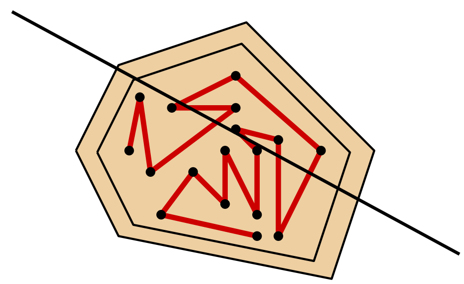


Gebt einen Algorithmus an, der (unter Zuhilfenahme des Algorithmus aus Teil 1) nach $\mathcal{O}(n)$ Vorberechnungszeit jedes Mal in $\mathcal{O}(\log(n) + k)$ Zeit die k Zutaten bestimmen kann, die auf der Ofen-Seite der Klinge liegen.

***Hinweis**: Ihr dürft annehmen, dass ihr als Eingabe die **verschachtelte konvexe Hülle** der Punkte $p_1, \dots, p_n$ bekommt. Diese erhält man, wenn man erst die konvexe Hülle aller Punkte und danach iterativ die konvexe Hülle der Punkte bestimmt, die nicht Teil einer vorherigen konvexen Hülle waren.*

Teilaufgabe (b) Eine letzte Baustelle in diesem (sonst optimalen) Vorgehen ist die Lebensdauer der Klinge, welche durch den Säuregehalt der Tomatensoße stark beeinträchtigt wird. Es soll nun ausgenutzt werden, dass die Tomatensoße auf der Pizza „verteilt" wird, indem ein Roboter die Soße in Form eines Polygonzuges, bestehend aus n Punkten, auf den Teig gibt. Wenn die Klinge zu viele Berührungen mit der Soße hätte, muss mittels Reset eine neue Ausrichtung probiert werden.

Gebt einen Algorithmus an, der (unter Zuhilfenahme des Algorithmus aus Teilaufgabe (a)) nach $\mathcal{O}(n \log n)$ Vorberechnungszeit in $\mathcal{O}((k+1)\log(n/(k+1)))$ Zeit die k Schnittpunkte der Klinge mit dem Polygonzug bestimmt.



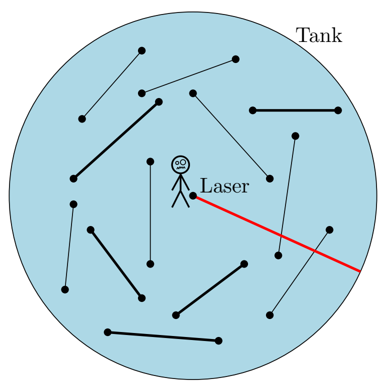
Aufgabe 2: Mission: Impossible **5 Punkte**

Die Luft wird knapp! Agentin Kim findet sich in einer prekären Situation wieder. Ihr Feind Dr. Meta hat sie mitten in einen zylinderförmigen Tank gesteckt, der komplett mit Wasser gefüllt ist. Zum Glück hat sie einen Laser (getarnt als Whiteboard-Marker), der die Wand des Tanks durchbrechen und sie retten könnte. Leider ist die Agentin von n Glasscheiben umgeben, deren Stärke die Kraft des Lasers beeinträchtigen.

Vor Panik rotiert die Agentin den Laser alle $\mathcal{O}(\log(n) + k)$ Sekunden zu einem zufällig gewählten Winkel, wobei k die Anzahl der Scheiben ist, die der Laser in der aktuellen Ausrichtung treffen würde. Nach einer Neuausrichtung muss Kim entscheiden, ob sie den Laser aktiviert. Da ein Laser in Whiteboard-Marker Form nur über eine sehr kleine Batterie verfügt, hat sie nur eine einzige Chance, um den Laser an der richtigen Stelle einzuschalten.

Zum Glück hatte Agentin Kim $\mathcal{O}(n \log n)$ Vorberechnungszeit und $\mathcal{O}(n)$ Speicher, in der sie zwar die Positionen der Glasscheiben studieren konnte, deren Stärke aber nicht! Für einen gegebenen Winkel muss sie nun die Summe der Stärken der getroffenen Scheiben bestimmen, um zu entscheiden, ob der Laser die Wand des Tanks erreichen würde.

Wie hat Kim die Vorberechnungszeit genutzt? Und wie kann sie nun schnell genug herausfinden, welche k Scheiben der Laser in einer gegebenen Position schneiden würde, bevor sie die Panik überkommt und sie zur nächsten Position rotiert?



Aufgabe 3: Anderthalbe Bereichsanfragen **5 Punkte**

Gegeben seien n Punkte in der Ebene. Eine anderthalbe Bereichsanfrage ist definiert durch $x_1, x_2, y \in \mathbb{R}$ und fragt die Teilmenge der Punkte an, die im Intervall $[x_1, x_2] \times [y, \infty]$ liegen. Gebt einen Algorithmus an, der nach $\mathcal{O}(n \log n)$ Vorberechnungszeit mit $\mathcal{O}(n)$ Speicher anderthalbe Bereichsanfragen in $\mathcal{O}(\log(n) + k)$ Zeit beantworten kann. Dabei ist k die Anzahl der Punkte, die im angefragten Bereich liegen.

Achtung: Es ist nur $\mathcal{O}(n)$ Speicher erlaubt!