גיאומטריה חישובית 09-08, פתרון תרגילי הבית:

להן פתרונות לתרגילי הבית שניתנו בקורס. אלו הם רק קווים כלליים לפתרונות, ואינם מכילים פירוט מלא והוכחות מפורטות (ניתן למצוא דוגמאות לפתרונות מפורטים במסמך הפתרונות לדוגמא). בנוסף, את רוב התרגילים ניתן לפתור בדרכים נוספות, השונות מאלו המתוארות כאן. במקרה של שאלות נוספות, בקשות, או טעויות בפתרונות, ניתן לשלוח מייל לאדם (הכתובת באתר הקורס).

תרגיל 1, שאלה 1:

פתרון מפורט לשאלה נמצא במסמך הפתרונות לדוגמא.

תרגיל 1, שאלה 2:

ניתן לפתור את שלושת סעיפי השאלה בצורה פשוטה ע"י איחוד של דיאגרמות נורמלים. ראינו בשיעור שניתן, בזמן לינארי בסיבוכיות הפוליגונים, למצוא את דיאגרמות הנורמלים שלהם, לסובב אותן, לאחד אותן, ואף לסרוק את האיחוד שלהן.

(a) שני הפוליגונים זרים אמ"מ קיים ישר המפריד ביניהם. ניתן להזיז ישר זה לכיוון כלשהו עד שיגע באחד הפוליגונים, ואז "לסובב" אותו לאורך מעטפת הפוליגון עד שיגע בפוליגון השני. כלומר, שני הפוליגונים זרים אמ"מ קיים ישר התומך בשניהם וכל פוליגון נמצא מצד אחר שלו.

אלג' שבודק האם קיים ישר כזה דומה מאוד לאלג' מציאת הגשרים שראינו בשיעור. ההבדל העיקרי בין שני האלגוריתמים הינו שהפעם צריך להשתמש בדיאגרמת הנורמלים ההפוכה (משוקפת סביב הראשית) של אחד מהפוליגונים, כיוון שאנו רוצים ששני הפוליגונים ישיקו לישר מכיוונים שונים.

(b) המרחק הקצר ביותר בין הפוליגונים מתקבל או בין זוג קודקודים, או בין קודקוד וצלע (חשוב להסביר מדוע לא ייתכן שהוא בין זוג צלעות!). בנוסף, קיים זוג ישרים מקבילים התומכים בפוליגונים בזוג הנקודות הקרובות ביותר, ואף נקודה השייכת לאחד מהפוליגונים אינה נמצאת ברצועה ביניהם (להסביר מדוע!).

מהפיסקה הקודמת נובעת נכונות האלג' הבא:

- ניצור את דיאגרמות הנורמלים של שני הפוליגונים.
- נסובב את הדיאגרמה של Q (לא באמת משנה איזו דיאגרמה בחרנו לסובב).
- נאחד את הדיאגרמה של P עם הדיאגרמה המסובבת של Q.
- כל קשת בדיאגרמה מתאימה לזוג קודקודים, אחד מP ואחד מQ. נבדוק את המרחק בין כל זוג כזה ונשמר את המרחק המינימאלי.

- בצורה דומה, כל קודקוד בדיאגרמה מתאים לצלע ולקודקוד. נבדוק גם את המרחק ביניהם.
- נחזיר את המרחק המינימאלי שמצאנו, ואת זוג הנקודות שמתאים לו.

(c) זהה לאלג' מהסעיף הקודם, אך הפעם אנו מחפשים את המרחק המקסימאלי. צריך להסביר מדוע המרחק המקסימאלי תמיד מתקבל בין זוג קודקודים (ולשנות את האלג' בהתאם. כלומר, למחוק את השלב לפני האחרון). בנוסף, צריך להסביר מדוע זוג הקודקודים עם המרחק הגדול ביותר מתאימים לכיווני נורמל הפוכים כלשהם (שוב להראות שקיים זוג ישרים מקבילים התומכים בשני הקודקודים, וכו'...).

תרגיל 1, שאלה 3:

(a) על מנת להראות קמירות של הסכום, צריך להוכיח שבין כל שתי נקודות שלו לא יכולה להיות נקודה שאינה בסכום. נניח בשלילה שהנקודות $s_1 = p_1 + q_1$ ו-$s_2 = p_2 + q_2$ נמצאות בסכום, אבל הנקודה $\alpha s_1 + (1-\alpha)s_2$ אינה נמצאת בסכום (כאשר $p_1, p_2 \in P$, $q_1, q_2 \in Q$, ועבור ערך כלשהו $0 < \alpha < 1$). נשים לב לכך שהנקודה $\alpha s_1 + (1-\alpha)s_2$ היא סכום של הנקודה $\alpha p_1 + (1-\alpha)p_2$ ושל הנקודה $\alpha q_1 + (1-\alpha)q_2$, אשר נמצאות בP וב-Q בהתאמה (נובע מכך ש-P ו-Q הינם קמורים). לכן, גם נקודה זו חייבת להיות בסכום, בסתירה להנחה.

כעת, על מנת להראות שמדובר בפוליגון, מספיק להראות שיש מספר קבוע של קודקודים במעטפת הקמור. זה נובע ישירות מסעיף c.

(b) נסובב את הצירים כך שכיוון u יהיה ציר ה-x. כעת קל לראות שהנקודה בסכום שמכילה ערך x מקסימאלי הינה הסכום של הנקודות משני הפוליגונים אשר מכילות את ערכי ה-x המקסימאליים בהם (לצורך הוכחה פורמאלית של שני הכיוונים, קל להשתמש בהנחה בשלילה).

(c) נאחד את דיאגרמות הנורמלים של P ושל Q. כל קשת בדיאגרמה החדשה מתאימה לזוג קודקודים משני הפוליגונים, אשר קיימת קבוצת כיוונים ששניהם קיצוניים בהם. לכן, מסעיף b מתקבל שכל קשת בדיאגרמה המאוחדת מתאימה לקודקוד של הסכום (ולהיפך). הקשת הזו מוכלת בתוך קשת של דיאגרמת הנורמלים של P ובתוך קשת נוספת של דיאגרמת הנורמלים של Q. הקודקוד המתאים לקשת בדיאגרמה המאוחדת נוצר על ידי סכימת שני הקודקודים המתאימים לקשתות המכילות אותה.

ישנם בדיוק $n+m$ קודקודים בדיאגרמת הנורמלים המאוחדת, ולכן זהו מספר הקודקודים אשר בסכום.

תרגיל 1, שאלה 4:

(a) ראינו בשיעור שאלגוריתם graham walk מוצא את הקמור של נקודות ממויינות בזמן לינארי. כיוון שהנקודות בקלט של השאלה ממויינות, ניתן למצוא את הקמור שלהן ב $O(n)$ זמן.

(b) ניתן בזמן לינארי לפרק את הנקודות שבקלט ל $k+1$ רשימות ממויינות על ידי מציאת k הנקודות הקיצוניות. נהפוך את הרשימות אשר ממויינות בסדר יורד (עדיין בזמן לינארי) ונקבל $k+1$ רשימות ממויינות בסדר עולה. איחוד כל הרשימות לרשימה ממויינת אחת יקח $O(n\log k)$ זמן, ושאר אלגוריתם מציאת הקמור יקח זמן לינארי (כמו בסעיף הקודם). לכן, ניתן למצוא את הקמור ב $O(n\log k)$ זמן.

תרגיל 1, שאלה 5:

נניח שקיים אלג' למציאת קמור של n נקודות אשר לוקח $CH(n)=o(n\log n)$ זמן, ונראה שגם מיון לוקח $o(n\log n)$ זמן (בסתירה להנחה שבשאלה). נראה אלגוריתם למיון n מספרים אשר רץ ב $O(n)+CH(n)=o(n\log n)$ זמן:

- נקבל n ערכי קלט - $a_1,a_2,...,a_n$.
- עבור כל ערך קלט a_i ניצור נקודה (a_i,a_i^2). זה יקח $O(n)$ זמן.
- נריץ את אלג' למציאת קמור אשר ירוץ בזמן $CH(n)$.
- נחזיר את הנקודות לפי הסדר שלהן על המעטפת התחתונה של הקמור, משמאל לימין. זה יקח $O(n)$ זמן.

מהגדרת הנקודות שאנו יוצרים, כל הנקודות ימצאו על הקמור התחתון, ממויינות לפי ערכי הx שלהן.

תרגיל 2, שאלה 1:

האלג' דומה מאוד לאלג' שראינו בשיעור עבור המקרה התלת מימדי. ההבדל העיקרי הינו שבמקום פרפרים אשר בנויים משני משולשים עם צלע משותפת, יהיו צורות אשר מורכבות משני טטראדרים עם משולש משותף (לצורך נוחות, נמשיך לכנות אותם בתור פרפרים). כמו באלג' המקורי, בכל שלב נוסיף נקודה (בסדר אקראי) ונבדוק אלו פרפרים צריך למחוק ואלו פרפרים צריך להוסיף. בנוסף, כמו באלג' המקורי נחזיק את רשימות שזוכרות אלו פרפרים כל נקודה שעוד לא התווספה רואה, וגם את הרשימות ההפוכות להן.

תוחלת מספר הפאות התלת מימדיות שהאלג' מייצר היא בעצם תוחלת מספר הפרפרים שהאלג' מייצר (פרט לחמש הנקודות הראשונות, כל פרפר מוסיף בדיוק פאה תלת מימדיות אחת). כמו באלג' המקורי, נגדיר את N_i להיות מספר הפרפרים במשקל i. לפי מה שראינו בשיעור, $N_0(n)=O(n^{\lfloor 4/2\rfloor})=O(n^2)$ (סיבוכיות הקמור של n נקודות בארבעה מימדים). לפי שיטת Clarkson-Shor, מתקבל $N_{\le w}(n)=O(w^d N_0(n/w))=O(w^5\bullet(n^2/w^2))=O(n^2w^3)$.

המשך החישוב דומה מאוד למקרה התלת מימדי שראינו בשיעור, ולבסוף אנו מגיעים לתוחלת של $O\left(\sum_w \frac{N_{\le w}(n)}{w^6}\right) = O\left(\sum_w \frac{n^2 w^3}{w^6}\right) = O\left(n^2 \sum_w \frac{1}{w^3}\right) = O(n^2)$.

תרגיל 2, שאלה 2:

(a) צלע e של הדיאגרמה מתאימה לצלע e' של הפוליהדרון, כאשר שתי הפאות ש-e מפרידה ביניהן בדיאגרמה מתאימות לשני הקודקודים ש-e' מחברת ביניהם בפוליהדרון. קודקוד v בדיאגרמה מתאים לפאה v' של הפוליהדרון, כאשר הפאות ש-v נמצאת על גבולן מתאימות לקודקודים של v'.

בהנחה שמדובר בפוליטופ, נקבל מהחישובים שעשינו בשיעור (בעזרת נוסחת אויילר) שיש בדיאגרמה $3n-6$ קשתות ו-$2n-4$ (גם כאשר לא מדובר בפוליטופ, לא יהיה שינוי בסדרי הגודל. למעשה, ניתן להראות חסמים עליונים למקרה של פוליהדרון אשר נבדלים מערכים אלו רק בקבוע קטן).

על מנת לבנות את הדיאגרמה, ניתן לעבור קשתות הפוליהדרון בסדר כלשהו (למשל, בעזרת DFS), ובכל שלב להוסיף את הקשת המתאימה לדיאגרמה. יש $O(n)$ קשתות בפוליהדרון, וטיפול בקשת יקח זמן קבוע. לכן, ניתן לבנות את הדיאגרמה בסיבוכיות זמן $O(n)$ (חסרים מספר פרטי מימוש שיש להסביר!).

(b) נטיל את הדיאגרמה על מישור, ונבצע עבורה הכנה לpoint location. לאחר מכן, עבור כל קודקוד v של הדיאגרמה (המייצג פאה v' בפוליהדרון), נחשב את הנקודה האנטיפודלית שלה p. נבצע point location עבור p על מנת למצוא איזו פאה מכילה את p. קודקוד הפוליגון אשר מתאים לפאה זו הינו הקודקוד הרחוק ביותר מהפאה v' (בדומה לשאלה 2 של תרגיל 1).

עיבוד מוקדם עבור point location יקח $O(n\log n)$, וכך גם ביצוע של n שאילתות. לכן, סיבוכיות זמן הריצה של האלג' הינה $O(n\log n)$ (להראות ששאר הפעולות באלג' יקחו זמן לינארי!).

(c) הטענה אינה נכונה, כיוון שייתכן שהעובי מתקבל בין זוג מישורים אשר תומכים בצלעות. דוגמא למקרה כזה ניתן למצוא בטטראדר משוכלל.

(d) מסעיף b נובע שניתן למצוא את המרחק המינימאלי בין זוג מישורים אשר אחד מהם תומך בפאה והשני תומך בקודקוד, בסיבוכיות זמן $O(n\log n)$. עם זאת, ייתכנו $O(n^2)$ זוגות של צלעות עם מישורים תומכים מקבילים. ניתן בזמן קבוע לבדוק האם לזוג צלעות יש מישורים תומכים מקבילים, ואם כן, למדוד את המרחק ביניהם. לכן, ניתן לבצע בדיקה זו עבור $O(n^2)$ הזוגות האפשריים ולמצוא את המרחק המינימאלי בין זוג מישורים אשר תומכים בצלעות של הפוליהדרון ומקבילים זה לזה. נשלב את שני החלקים (פאה-קודקוד וצלע-צלע) ונקבל אלג' למציאת עובי אשר סיבוכיות הזמן שלו הינה $O(n^2)$.

תרגיל 2, שאלה 3:

נסמן את נקודות הקלט בתור $p_1, p_2, ..., p_n$. נהפוך כל נקודה p_i לריבוע S_i שצלעו a ומרכזו בנקודה (נכנה ריבועים אלו בשם ריבועי קלט. לחילופין, ניתן ליצור ריבוע שהנקודה היא אחת מפינותיו). נבחן נקודה p' שאינה נקודת קלט והריבוע המתאים לה הינו S'. נשים לב ש p' נמצאת בתוך ריבוע קלט S_i אמ"מ נקודת הקלט p_i נמצאת בתוך הריבוע S'. כלומר, מציאת הריבוע שמכיל מספר מקסימאלי של נקודות קלט שקולה למציאת הנקודה שנמצאת בתוך מספר המקסימאלי של ריבועי קלט.

נמצא את הנקודה המקסימאלית באמצעות ביצוע sweep על מערך ריבועי הקלט. האירועים של הsweep יהיו נקודות התחלה ונקודות סיום של ריבועים. מבנה ה-y יחזיק את כל התאים במערך שישר הsweep חוצה באותה נקודה. בנוסף, לכל תא שנמצא במבנה ה-y, נשמור את העומק שלו (מספר הריבועים שהוא מוכל בהם). על מנת למצוא את התא עם העומק המקסימאלי, נחזיק את העומק המקסימאלי שנתקלנו בו עד כה, ועבור כל תחילת תא נבדוק האם התא החדש עמוק יותר.

כעת נבחן מהי סיבוכיות זמן הריצה של האלג':

- הפיכת כל נקודת קלט לריבוע ואתחול מבנה ה-x בהתאם - $O(n \log n)$.
- קיימים $2n$ אירועים. טיפול באירוע יכול לקחת $O(n)$ זמן כיוון שהוא עלול ליצור $O(n)$ תאים חדשים (וצריך למשל לשמור את העומק של כל אחד מהם).

לסיכום, סיבוכיות זמן הריצה של האלג' הינה $O(n^2)$.

שימו לב שחסר פירוט של הפעולות שיש לבצע בכל אירוע, וגם בחישוב הסיבוכיות של כל פעולה כזו.

תרגיל 2, שאלה 4:

האלג' יבצע sweep שאירועיו הינם קודקודי הפוליגונים. מבנה ה-y יחזיק את הצלעות האופקיות שישר הsweep חותך באותו זמן. כל אירוע מתאים להתחלה/סיום של שני קטעים אופקיים, ולכן הקשתות המתאימות יוצאו/יוכנסו למבנה ה-y. בנוסף, כל אירוע מתאים לקטע אנכי כלשהו, ויש לבדוק כמה קטעים אופקיים הוא חותך. לצורך כך, נמצא במבנה ה-y את הקטע הראשון הנמצא מתחת לקצה העליון של הקטע האנכי, ואת הקטע הראשון הנמצא מעל לקצה התחתון שלו (לא כולל קטעים שהוכנסו למבנה באירוע זה). לאחר מכן נבדוק כמה קטעים במבנה ה-y נמצאים בין שני הקטעים שמצאנו (למשל, בעזרת עץ מאוזן שתומך בorder statistics). נחזיק מונה שיספור כמה חיתוכים מצאנו עד כה, ובכל אירוע נוסיף למונה את מספר הקטעים שהקטע האנכי הנוכחי חותך.

נבחן מהי סיבוכיות זמן הריצה של האלג':

- תחילה, יצירת מבנה ה-x תדרוש מיון של $n/2$ ערכים, ולכן תקח $O(n \log n)$.

- ישנם בדיוק $n/2$ אירועים. בכל אירוע מתבצעות מספר קבוע של פעולות הכנסה למבנה ה-y, הוצאה ממבנה ה-y, ומציאת מספר הקטעים במבנה שנחתכים על ידי קטע אנכי. כיוון שגודלו של מבנה ה-y הינו לכל היותר $n/2$, כל אירוע יקח $O(\log n)$ זמן, ולכן, סך הזמן שיקח לטפל בכל האירועים הינו $O(n\log n)$.

לסיכום, סיבוכיות זמן הריצה של האלג' הינה $O(n\log n)$.

כעת נעבור לחלק השני של השאלה ונניח ש-Q הינו סט של קטעים זרים במישור. נבצע מספר שינויים באלג' על מנת שיספור את מספר החיתוכים בין הקטעים האנכיים של P לבין קטעי Q. נריץ את האלג', ולאחר מכן נסובב את הצירים כך שהקטעים האופקיים של P יהפכו להיות אנכיים, ונחזור על האלג'. נחבר את מספר החיתוכים שקיבלנו בשתי ריצות האלג', ונחזיר אותו.

האירועים באלג' החדש יהיו נקודות הקצה של קטעי Q וערכי ה-x אשר בהם נמצא קטע אנכי של P. באירוע מהסוג הראשון, נעדכן את מבנה ה-y בהתאם (נכניס או נוציא את הקטע המתאים). באירוע מהסוג השני, נספור את מספר החיתוכים בין הקטע האנכי לבין הקטעים שבמבנה ה-y (באותה צורה כמו באלג' המקורי). בדיקה קצרה מגלה ששינויים אלו אינם משפיעים על זמן הריצה של האלג'.

תרגיל 2, שאלה 5:

פתרון מפורט לשאלה נמצא במסמך הפתרונות לדוגמא.

תרגיל 3, שאלה 1:

מכל קודקוד של קטע נעביר ישר המקביל לציר ה-y. בצורה כזו, חילקנו את המישור ל $2n-1$ רצועות (ועוד שני חצאי מישורים ריקים), כך שבתוך כל רצועה לא מתחיל/מסתיים אף קטע. נוכל להחזיק לכל רצועה עץ חיפוש מאוזן שמחזיק את הקטעים ברצועה ממויינים לפי ציר ה-y. נוסיף לכל עץ תמיכה בחישוב מהיר של מספר הקטעים מתחת לקטע מסויים (למשל, עץ עם order statistics, אשר מאפשר בדיקת מידע כזה בזמן לוגריתמי, על ידי שמירת מספר הצאצאים עבור כל איבר בעץ). כאשר תתקבל שאילתא, נמצא את הרצועה המתאימה לקטע הקלט האנכי, נמצא בעץ המתאים לרצועה את הקטע העליון ביותר שהקטע האנכי חותך, ואת הקטע התחתון ביותר שהוא חותך. לסיום, נבדוק כמה קטעים נמצאים בין שני הקטעים שמצאנו (לצורך כך נשתמש במספר הצאצאים של כל קטע בעץ, ששמרנו קודם).

כל אחד משלבי השאילתא באלג' שתארנו לוקחים $O(\log n)$ זמן, ולכן השאילתא מתבצעת בזמן לוגריתמי. עם זאת, תחזוק של $O(n)$ עצים יקח $O(n^2)$ מקום (וגם זמן רב של עיבוד מוקדם). נשים לב שבין עצים של שתי רצועות עוקבות יש הבדל קבוע, ולכן נוכל להשתמש בpersistence.

במקרה זה, יש בעיה עם השימוש בpersistent search tree, כיוון שאין דרך פשוטה לשלב אותו עם order statistics (זה ידרוש מכל איבר בעץ לשמור את מספר הבנים שלו בכל גירסא!).

לכן, נשתמש בpath copying. זה יגרום לסיבוכיות מקום וסיבוכיות זמן עיבוד מוקדם של $O(n\log n)$.

תרגיל 3, שאלה 2:

אלג' השאילתא יבצע שתי בדיקות עבור q:

- האם q נמצאת מתחת למעטפת העליונה של הקמור המתאים.
- האם q נמצאת מעל למעטפת התחתונה של הקמור המתאים.

האלג' יחזיר תשובה חיובית אמ"מ שתי הבדיקות חזרו עם תשובה חיובית. נראה כיצד מבצעים את הבדיקה הראשונה, ואילו את הבדיקה השניה ניתן לבצע באופן דומה.

נבנה עץ חיפוש מאוזן שיכיל את כל הנקודות ממויינות לפי ערך ה-x שלהן. בצורה כזו, בהנתן a, נוכל למצוא בזמן לוגריתמי את הנקודה עם ערך ה-x המקסימאלי שעדיין קטן מ-a (כלומר, את הנקודה הימנית ביותר של הקמור הרלוונטי). נבנה את הקמור העליון בשיטת graham walk שראינו בשיעור, אך נשמור את גירסאות הביניים של המעטפת שנוצרו לאחר ביצוע כל שלב. עבור כל נקודה בעץ שבנינו קודם לכן, נשמור מצביע למעטפת העליונה שנוצרה ע"י האלג' מיד לאחר הוספת הנקודה.

בהנתן שאילתא, נשתמש ב-a כדי למצוא את המעטפת העליונה המתאימה, ונחפש בה את הצלע שמכילה את ערך ה-x של q (אם לא קיימת צלע כזו, ברור שהנקודה q אינה בקמור). לאחר מכן, נבדוק האם q נמצאת מתחת לצלע שמצאנו (היא נמצאת מתחת לצלע זו אמ"מ היא נמצאת מתחת לקמור העליון).

על מנת לחסוך במקום ובזמן העיבוד המוקדם, נרצה לזרוק את עץ המעטפות שבנינו ולהשתמש בpersistent search tree שיחזיק את המעטפות. אם נבחר את חותמת הזמן של מעטפת להיות ערך ה-x של הנקודה הימנית שלה, אנו עלולים לקבל שני קמורים עם חותמות זמן סמוכות ו $O(n)$ הבדלים ביניהם (במקרה ובדיוק הוסרו $O(n)$ קודקודים מהקמור). על מנת להתגבר על בעיה זו, ניצור גירסא של הקמור לאחר כל שינוי שהאלג' יבצע בו (הוספה או הסרה של קודקודים). נקבל לכל היותר $2n-2$ גרסאות, כאשר המעטפות של כל שתי גרסאות סמוכות נבדלות בקודקוד יחיד (כמובן שלא נתעניין בחותמות הזמן של גרסאות הביניים החדשות).

השאילתא כוללת חיפוש בpersistent search tree ומספר בדיקות שניתן לבצע בזמן קבוע. לכן, סיבוכיות זמן השאילתא הינה $O(\log n)$. סיבוכיות המקום הינה $O(n)$ וסיבוכיות זמן העיבוד המקדים הינה $O(n\log n)$ (צריך להסביר במפורט את התוצאות האלו).

תרגיל 3, שאלה 3:

נטיל את כל המשולשים על מישור xy (למשל, על המישור $z=0$), ונקבל מפה מישורית המכילה $O(n+k)$ פאות. נעבד את המפה לpoint location כמו שראינו בשיעור (זה לא ישפיע על

סיבוכיות מספר פאות). נשים לב שבתוך פאה לא מתחיל/מסתיים אף משולש. לכן, ניתן להחזיק לכל פאה עץ חיפוש מאוזן שיכיל את כל המשולשים המכילים את הפאה, ממויינים לפי גובהם.

בהינתן נקודת שאילתא, נוכל למצוא במישור הxy את הפאה המכילה אותה (באמצעות שאילתת point location), ולאחר מכן למצוא את המשולש שנמצא ישירות מתחת לנקודה בעזרת חיפוש בעץ המתאים לפאה. מציאת הפאה המתאימה במפה המישורית תיקח $O(\log(n+k)) = O(\log n)$ זמן, וחיפוש בעץ יקח $O(\log n)$. כלומר, זמן ביצוע שאילתא הינו לוגריתמי ב-n. עם זאת, ביצוע נאיבי של האלג' ידרוש מקום רב וזמן עיבוד מוקדם רב.

ננצל את העובדה שבין שני עצים המתאימים לשתי פאות סמוכות במבנה הנתונים קיים הבדל קבוע (העצים זהים, פרט למשולש בודד שנמצא רק באחד מהם). נרצה להשתמש ב persistent search tree, אך הגדרת הtimestamp שלו איננה טריוויאלית במקרה זה. נבנה גרף שקודקודיו מייצגים את פאות המפה המישורים, וקיימת קשת בין שני קודקודים סמוכים אמ"מ לפאות שלהם יש צלע משותפת. נמצא עץ פורש בגרף, נהפוך כל אחת מקשתות העץ לשתי קשתות מכוונות, ונמצא מסלול אויילר בעץ המכוון. קיבלנו מסלול העובר בכל הפאות, מכיל $2(n-1)$ צעדים, ולכל שני איברים סמוכים במסלול יש עצים שההבדל ביניהם קבוע. לכן, נוכל לבחור את המיקום של פאה במסלול להיות הtimestamp שלה (ביקור חוזר בפאה יצור timestamp חסר משמעות עבורנו, אך סיבוכיות העץ לא תושפע מכך).

קל לראות שהשינוי הנ"ל לא פגע בזמן השאליתא. סיבוכיות המקום הינה $O(n+k)$, וסיבוכיות זמן העיבוד המוקדם הינה $O((n+k)\log n)$ (צריך לפרט מדוע!).

<u>תרגיל 3, שאלה 4:</u>

לכל מישור קלט, קיימים שני מישורים אשר מקבילים לו ותומכים בפוליטופ הנתון (כל מישור תומך בפוליטופ מצד אחר שלו). מישור הקלט חותך את הפוליטופ אמ"מ הוא נמצא בין שני המישורים האלו. בעיבוד המוקדם נבצע את הפעולות הבאות:

- נבנה את דיאגרמת הנורמלים של הפוליטופ ונטיל על מישור על מנת לקבל מפה מישורית. בשאלה 2 של תרגיל 2 ראינו שניתן לבצע זאת כזו ב $O(n)$ סיבוכיות זמן ומקום.
- נעבד את המפה המישורית לשאילתות point location (כמו שראינו בשיעור). סיבוכיות המקום תישאר לינארית, אך סיבוכיות זמן העיבוד המוקדם תגדל ל $O(n\log n)$.

בהנתן מישור שאילתא, נבדוק מהם שני הכיוונים המאונכים למישור, ונחפש במפה המישורית את שתי הפאות המכילות את הנקודות המתאימות לכיוונים אלו. כל אחת משתי הפאות האלו מתאימה לקודקוד של הפוליגון, אשר קיים מישור תומך אשר עובר דרכו ומקביל למישור הקלט. נבדוק האם כל אחד משני הקודקודים שקיבלנו נמצא בצד שונה של המישור (כלומר, ששני הקודקודים אינם נמצאים באותו חצי מישור הנוצר ע"י המישור), ונשיב מהתאם.

השאילתא כוללת שני חיפושים בזמן לוגריתמי, ועוד מספר בדיקות שניתן לבצע בזמן קבוע (בדיקת היחס בין שני הקודקודים למישור). לכן, השאילתא מתבצעת בזמן $O(\log n)$. סיבוכיות המקום הינה $O(n)$, וסיבוכיות זמן העיבוד המוקדם הינה $O(n\log n)$.

תרגיל 3, שאלה 5:

(a) נחלק את המישור ל $O(n^2)$ רצועות (ושני חצאי מישורים) על ידי העברת ישר מקביל לציר ה-y מכל קודקוד במפה, בדומה לשאלה 1 של תרגיל 3 (גם המשך הפתרון כמעט זהה לפתרון שאלה זו). לכל רצועה ניתן לשמור עץ חיפוש מאוזן שיחזיק את הישירים ממויינים לפי הסדר שלהם ברצועה. כל ישר בכל עץ יזכור כמה ישרים נמצאים מתחתיו ברצועה המתאימה. ניתן לבנות את כל הרצועות בעזרת sweep, אשר אירועיו הם נקודות החיתוך של הישרים, ובכל אירוע צריך לעדכן בדיוק שני ישרים. אתחול מבנה ה-y יתבצע על ידי מיון הישרים לפי שיפועם.

בין עצים של רצועות עוקבות יש מספר שינויים קבוע, ולכן, במקום להחזיק כל עץ בנפרד, ניתן להשתמש בpersistent search tree. כיוון שיש $O(n^2)$ גירסאות של העץ (כלומר, $O(n^2)$ רצועות) וכל גירסא לוקחת מקום לינארי, סיבוכיות גודל העץ הינה $O(n^2)$ וסיבוכיות זמן העיבוד המוקדם שבנייתו תיקח הינה $O(n^2 \log n)$ (צריך להראות שכל הפעולות שביצענו אכן לוקחות זמן כזה). שאילתא תמצא את הישר שנמצא ישירות מתחתיה בגרסת העץ המתאימה, והישר בעץ יכיל גם את מספר הישירים שתחתיו בגירסה זו (כלומר, מספר הישרים אשר מתחתיו ברצועה זו). לכן, סיבוכיות זמן השאילתא הינה $O(\log n)$.

(b) נחלק את הישרים לקבוצות בגודל $\sqrt{m}$. חישוב פשוט מראה שיהיו $O\left(\frac{n}{\sqrt{m}}+1\right)$ קבוצות כאלו (ההסופה של 1 נדרשת עבור המקרים בהם $\sqrt{m} > n$). לכל קבוצה כזו נבצע עיבוד מוקדם לפי הסעיף הקודם, ובהניתן נקודת שאילתא, נריץ אותה על כל אחת מהקבוצות ונסכום את התוצאות שיתקבלו. סכימה זו תביא אותנו למספר הישרים אשר מתחת לנקודה.

סיבוכיות העיבוד המוקדם של קבוצה אחת הינה $O(m \log m)$, ולכן סיבוכיות זמן העיבוד המוקדם הכוללת הינה $O(m \log m) \bullet O\left(\frac{n}{\sqrt{m}}+1\right) = O\left(n\sqrt{m} \bullet \log m + m \log m\right)$.

סיבוכיות המקום שכל קבוצה דורשת הינה $O\left(\sqrt{m}\right)$, ולכן סיבוכיות המקום הכוללת הינה $O(m) \bullet O\left(\frac{n}{\sqrt{m}}+1\right) = O\left(n\sqrt{m}+m\right)$.

סיבוכיות הזמן של שאילתא בקבוצה אחת הינה $O(\log m)$, וסיבוכיות הזמן של סכימת התוצאות של כל הקבוצות הינה $O\left(\frac{n}{\sqrt{m}}+1\right)$. לכן, סיבוכיות הזמן הכוללת של שאילתא הינה $O\left(\left(\frac{n}{\sqrt{m}}+1\right)\log m\right)$. מכך נובע שסיבוכיות הזמן של כל m השאילתות הינה $O\left(n\sqrt{m} \bullet \log m + m \log m\right)$, וזוהי גם סיבוכיות הזמן הכוללת של העיבוד המקדים והשאליתות.

(c) נשתמש בשיטת ה-doubling. תחילה ננחש ש-m הוא ערך קבוע כלשהו (למשל 2), ובכל פעם שמספר השאילתות יעבור את הערך הנוכחי של m, נכפיל את הערך ב-2. ההערכה הסופית עבור m יכולה להיות לכל היותר $2(m-1)$.

נבחן את סיבוכיות סך העיבודים המוקדמים שיתבצעו בתהליך. בצורה דומה ניתן לחשב את סיבוכיות השאילתות וסיבוכיות המקום.

סדרת העיבודים המוקדמים תתבצע עבור הערכים $2, 2^2, 2^3, ..., 2^{\log(2m-1)}$ (ייתכן שהאיבר האחרון קטן יותר). מהסעיף הקודם, ידוע לנו שזמן העיבוד המוקדם עבור m כלשהו חסום על ידי $cn\sqrt{m}\bullet\log m + cm\log m$ (עבור קבוע c כלשהו). כלומר, הזמן הכולל של העיבודים המוקדמים חסום על ידי

$$
\begin{aligned}
& c\left[n\sqrt{2}\bullet\log 2 + 2\log 2\right] + c\left[n\sqrt{4}\bullet\log 4 + 4\log 4\right] + ... + \\
& + c\left[n\sqrt{(2m-1)}\bullet\log(2m-1) + (2m-1)\log(2m-1)\right] = \\
& = cn\sum_{i=1}^{\log(2m+1)}\sqrt{2^i}\bullet i + c\sum_{i=1}^{\log(2m+1)}2^i\bullet i \le \\
& \le cn\log(2m-1)\sum_{i=1}^{\log(2m+1)}2^{i/2} + c\log(2m-1)\sum_{i=1}^{\log(2m+1)}2^i \le \\
& \le cn\log(2m-1)2^{1+0.5\log(2m+1)} + c\log(2m-1)2^{1+\log(2m+1)} = O(n\sqrt{m}\log m + m\log m)
\end{aligned}
$$

<u>תרגיל 4, שאלה 1:</u>

(a) נסמן את קבוצת הנקודות בתור $P = \{p_1, p_2, ..., p_n\}$, ונקודה p_i תסומן בתור $p_i = \left\{x_1^i, x_2^i, ..., x_d^i\right\}$.

נקודה נמצאת ב-C אמ"מ קיים על-מישור אשר מפריד בינה לבין $n-1$ הנקודות הנוספות. נרצה להשתמש בבעיית Linear Programming על מנת לבדוק האם קיים על מישור כזה, התומך בנקודה p_s (כאשר $1 \le s \le n$). המשתנים של בעיית ה-LP יהיו המקדמים המגדירים את העל-מישור הנ"ל $a_1x_1 + a_2x_2 + ... + a_dx_d = b$ (כלומר, ישנם $d+1$ משתנים, $a_1, a_2, ..., a_d, b$). האילוצים יהיו:

$$
\begin{aligned}
& \forall i \ne s : a_1p_1^i + a_2p_2^i + ... + a_dp_d^i \le b \\
& a_1p_1^s + a_2p_2^s + ... + a_dp_d^s \ge b
\end{aligned}
$$

לא משנה באיזו פונקציית מטרה נשתמש, כיוון שמספיק לנו לדעת האם קיימת תוצאה כלשהי למישוואות האלו (רצוי להסביר מדוע מספיק להשתמש בפעולת קטן-שווה ולא קטן-ממש).
כלומר, הנקודה p_s בקמור C אמ"מ התקבל פתרון לבעיה. ישנם $d+1$ משתנים ו- n אילוצים, ולכן ניתן לפתור את בעיית ה-LP בסיבוכיות זמן $O(n)$. נריץ אלג' LP עבור כל אחת מ- n נקודות הקלט ונקבל סיבוכיות זמן $O(n^2)$.

(b) נשים לב שפאה j-מימדית מורכבת מ $j+1$ קודקודים. נעבור על כל $\binom{n}{j+1}=O\left(n^{j+1}\right)$ הפאות האפשריות, ונבדוק האם כל אחת מהן נמצאת על מעטפת הקמור.

נראה כיצד מתבצעת בדיקה עבור הפאה הבנויה מקבוצת הקודקודים $F=\{q_1,q_2,...,q_{j+1}\}$, כאשר $\forall i: q_i \in P$ (ואין איברים המופיעים יותר מפעם אחת בקבוצה). בדומה לסעיף a, הקבוצה מייצגת פאה j-מימדית של C אמ"מ קיים על-מישור אשר מפריד בין F לבין P/F. לכן, שוב נשתמש בבעית תכנות לינארי שתבדוק האם קיים מישור כזה. הפעם המשוואות יהיו:

$$i \in F: a_1 p_1^i + a_2 p_2^i + ... + a_d p_d^i \le b$$
$$i \notin F: a_1 p_1^s + a_2 p_2^s + ... + a_d p_d^s \ge b$$

קיבלנו בעיית LP עם n אילוצים המשתמשים ב $d+1$ משתנים, ולכן סיבוכיות זמן הריצה של הבדיקה הינה $O(n)$. סיבוכיות זמן הריצה של כל הבדיקות הינה $O\left(n^{j+1}\right)\bullet O(n)=O\left(n^{j+2}\right)$.

<u>תרגיל 4, שאלה 2:</u>

(a) ישר חותך משולש אמ"מ הוא חותך לפחות את אחד משלושת הקטעים המרכיבים אותו (למעשה, הוא לא יכול לחתוך קטע אחד בדיוק, אבל זה לא משנה לנו). ראינו בשיעור שקטע במישור הפרימאלי הופך ל-double wedge במישור הדואלי. לכן, ישר l חותך משולש t במישור הפרימאלי אמ"מ l^* נמצא בלפחות אחד משלושת הdouble wedges שמייצגים את צלעות t.

נסתכל על מערך הישרים שנוצר על ידי ה-DWs במישור הדואלי. זהו מערך של $O(n)$ ישרים, ולכן סיבוכיותו $O(n^2)$. נגדיר את הדרגה של תא במערך להיות מספר הDouble Wedges שהוא מוכל בהם (כלומר, אנו רוצים לדעת האם קיים תא עם דרגה 0). נבנה את המערך על ידי אלג' sweep סטנדרטי לבניית מפה מישורית מקבוצה של ישרים. לכל ישר במבנה ה-y נשמור מהי דרגת התא אשר נמצא ישירות תחתיו (בכל אירוע נצטרך לעדכן מספר קבוע של ערכים כאלו). אם במהלך הsweep נגיע לתא שדרגתו 0, והוא אינו התא הנמוך ביותר, מצאנו ישרים מפרידים.

צריך לפרט כיצד מאתחלים את מבנה ה-y, ואילו פעולות יש לעשות כאשר מגיעים לאירוע. סיבוכיות הsweep אינה משתנה עקב פעולות אלו, ולכן היא נשארת $O(n^2 \log n)$.

(b) ראינו בשיעור שקבוצה קמורה ללא קודקודים הופכת במישור הדואלי למסדרון ששני "הקירות" שלו הינם עקומים קמורים. במקרה שמדובר בדיסקים, אלו הם עקומים פשוטים, ולכן ניתן לבצע עליהם פעולות בזמן קבוע.

נבצע את אותו ה-sweep שביצענו בסעיף הקודם, אך הפעם על מערך של $O(n)$ עקומים. כיוון ששני עקומים פשוטים נחתכים מספר קבוע של פעמים, סיבוכיות המפה המישורית הנוצרת תשאר $O(n^2)$, וזמן הריצה של האלגוריתם לא ישתנה.

(c) נתחיל מהמקרה של המשולשים. נשים לב שקיים ישר מפריד אמ"מ קיים ישר מפריד אשר עובר דרך לפחות קודקוד אחד של משולש (כיוון שניתן להזיז את הישר המפריד לכיוון כלשהו, עד שיגע במשולש כלשהו). לכן, מספיק לבדוק האם קיים ישר מפריד העובר דרך קודקוד של משולש כלשהו.

עבור כל קודקוד של כל משולש, נבצע sweep מעגלי ממנו (קרן הsweep תמיד עוברת דרך הקודקוד ומסתובבת. היא תתחיל מלתמוך בצלע אחת של המשולש, ותסתובב עד שתתמוך בצלע אחרת שלו). במהלך הsweep נתחזק את מספר המשולשים שהקרן חותכת באותו רגע, ולכן, האירועים יהיו נקודות ההשקה של הקרן עם המשולשים האחרים. צריך למצוא זוג קרניים הפוכות (כלומר, שתי קריין אשר מוכלות באותו ישר, אך פונות לכיוונים שונים שלו) אשר אינן חותכות אף משולש (צריך לפרט איך בדיוק מגלים מקרה כזה, ואיך בודקים שהישר שהתקבל אינו מעל/מתחת לכל המשולשים).

בכל sweep יש $O(n)$ אירועים, ולא צריך לתחזק כלום במבנה ה-y. לכן, סיבוכיות הזמן של כל sweep הינה $O(n\log n)$. מבצעים $O(n)$ פעולות sweep, ולכן הסיבוכיות הכוללת הינה $O(n^2\log n)$.

הטיפול במקרה של הדיסקים דורש שינויים קלים בלבד. נבצע sweep אחד עבור כל דיסק, כך שישר ה-sweep משיק לדיסק ומסתובב סביבו (כלומר, נקודת ההשקה נעה לאורך הדיסק). אירוע יתרחש כאשר ישר ה-sweep משיק לדיסק נוסף. כיוון שלכל שני דיסקים יש שני משיקים משותפים (פרט למקרה הלא מעניין בו דיסק מוכל בדיסק אחר), יהיו $O(n)$ אירועים בכל sweep, והסיבוכיות הכוללת תשאר $O(n^2\log n)$.

תרגיל 4, שאלה 3:

נסמן את n קטעי הקלט באמצעות נקודות הקצה שלהם $[a_1,b_1],[a_2,b_2],...,[a_n,b_n]$. הסימונים $a_i.x$ ו- $a_i.y$ יסמנו את הקואורדינטות של הנקודה a_i. למשל, כיוון שהקטעים אופקיים, מתקיים $\forall i: a_i.y = b_i.y$.

האלג' יגדיר בעיית Linear Programming ויפתור אותה. המשתנים הם c,d, והם מגדירים ישר $y = cx + d$. פונקציית המטרה הינה $\max(a)$ (על מנת למצוא את הישר עם השיפוע המקסימאלי). לצורך נוחות, נגדיר את הישר בתור $x = \frac{y-d}{c}$ (יש להסביר בנפרד כיצד מתמודדים עם מקרים בהם הישר הרצוי הינו מאוזן או מאונך). נרצה שהאילוצים יהיו

$$\forall i: a_i.x \le \frac{a_i.y - d}{c}$$
$$\forall i: b_i.x \ge \frac{b_i.y - d}{c}$$

אך אלו אינם אילוצים לינאריים, ולא קל להפוך אותם לכאלו, כיוון שאין אנו יודעים האם c חיובי או שלילי. לכן, נפצל לשתי בעיות LP – הבעיה הראשונה תבדוק האם קיים ישר עם שיפוע חיובי, ובמידה ואין כזה, נבצע בעיית LP נוספת שתבדוק האם קיים ישר עם שיפוע שלילי. לבעיה

הראשונה נוסיף את האילוץ $c \geq 0$. הבעיה השניה זהה לבעיה הראשונה פרט לכך שיש להפוך את כיווני האי-שיוויונים בכל האילוצים.

קיבלנו שתי בעיות LP, כל אחת עם שני משתנים ו- $2n+1$ אילוצים. האלג' יבנה את הבעיות ויפתור אותה בסיבוכיות זמן $O(n)$. אם אין פתרון לשתי הבעיות, לא קיים ישר אשר חותך את כל הקטעים. אחרת, הישר המבוקש יוחזר כפתרון לאחת הבעיות.

תרגיל 4, שאלה 4:

(a) ניקח ישר l אשר עובר מעל נקודת קלט אחת, ודרך לפחות נקודת קלט אחת. נשתמש בדואליות המשמרת יחסי מעל/מתחת (בשיעור התייחסנו אליה בתור "הדואליות השניה"). l^* הינה נקודה (במישור הדואלי) אשר לפחות ישר אחד של P^* עובר דרכה, ובדיוק ישר אחד נמצא מעליה.

נגדיר את הדרגה של נקודה במישור הדואלי להיות מספר הישרים העוברים מעליה. בנוסף, עבור ערך x כלשהו, נגדיר את סדר ישרי P^* להיות מהגבוה לנמוך (כאשר הישר העליון הינו הראשון בסדר). L_1^* הינה קבוצה של קטעים בעלי דרגה 1. בנוסף, כל קטע נמצא על ישר כלשהו $q^* \in P^*$, בין שתי נקודות חיתוך עוקבות שלו. היות שלקטע יש דרגה 1, q^* הינו הישר השני בסדר עבור ערכי ה-x של הקטע. לכן, בנקודות החיתוך שמייצגות את קצוות הקטע, הוא נחתך עם הישר השלישי בסדר, או עם הישר הראשון בסדר. נשים לב שרק במקרה הראשון נקודת הקצה של הקטע הינה חלק מ- L_1^*. לכן, L_1^* הינו מסלול x-מונוטוני של קטעים בדרגה 1 (אשר כל אחד מהם שוכן על ישר של P^*, בין שתי נקודות קצה סמוכות שלו), מינוס נקודות הקצה שהינן חיתוכים בין ישר מספר 1 וישר מספר 2.

(b) מספר הקודקודים ב- L_1^* חסום (אסימפטוטית) על ידי מספר הקטעים בו, ולכן מספיק לחסום את מספר הקטעים. נסמן את מספר הקטעים מדרגה i בתור $N_i(n)$. קל לראות ש $N_0(n) = O(n)$ (להסביר מדוע כל ישר מופיע לכל היותר פעם אחת על המעטפת העליונה). כל קטע מוגדר על ידי שלושה ישרים (הקרניים בקצוות מוגדרות ע"י שני ישרים), ולכן מקלרקסון-שור נובע $N_1(n) < N_{\leq 1}(n) = O\left(2^3 \bullet N_0(n/1)\right) = O(n)$.

תרגיל 4, שאלה 5:

(a) נחלק את הבעיה לשתי בעיות סימטריות אשר נפתור באותה צורה. הבעיה הראשונה מניחה ש-P נמצא מעל Q, ואילו הבעיה השניה מניחה ש-Q נמצא מעל P. כל בעיה תמצא מהו המרחק שיש להוריד את הפוליטופ "העליון" עד שישיק לתחתון. במידה והורדה של הפוליטופ "העליון" לעולם לא תגרום לו לגעת בפוליטופ התחתון, יוחזר ערך אינסופי. כיוון שמדובר בפוליטופים קמורים, לפחות אחת מהבעיות חייבת להחזיר ערך אינסופי. אם אחת הבעיות החזירה ערך

סופי, זהו המרחק הרצוי. אחרת, אף פוליטופ לא נמצא ישירות מעל השני. נסביר כיצד לפתור את הבעיה הראשונה.

נזיז את P כלפי מטה עד שישיק ל-Q, ונסמן את המרחק ש-P עבר ב- h. כיוון שמדובר בפוליטופים קמורים, ניתן להעביר מישור אשר עובר דרך נקודת ההשקה, ושכל אחד משני חצאי המרחבים שהוא משרה מכיל את אחד הפוליטופים. זאת אומרת שקיימים זוג מישורים מקבילים במרחק h זה מזה, כך שהראשון תומך ב-P מלמטה והשני תומך ב-Q מלמעלה. אם נוריד מישור התומך ב-P מלמטה במרחק הגדול מ- h, הוא יחתוך את Q או שיהיה מתחתיו. לכן, לא ייתכן זוג מישורים מקבילים כאלו שהמרחק האנכי ביניהם גדול מ- h. כלומר, מספיק למצוא את זוג המישורים המקבילים, שהראשון מביניהם תומך ב-P מלמטה והשני תומך ב-Q מלמעלה, והמרחק האנכי ביניהם מקסימאלי (צריך להתייחס גם למקרה בו P אינו נמצא מעל Q). נמצא את זוג המישורים בעזרת Linear Programming שמשתניה הם המקדמים שלהם:

$$z = ax + by + c$$
$$z = ax + by + c'$$

הישר העליון יהיה הישר שתומך ב-P. יש לנו ארבע משתנים בבעיה - a,b,c,c'. כיוון שאנו מניחים ש-P נמצא מעל Q, נרצה להגדיר את האילוץ $c \geq c'$, ופונקציית המטרה תהיה $\max(c-c')$. נגדיר את P',Q' להיות קבוצות הקודקודים של הפוליטופים. האילוצים יהיו

$$\forall (x_i, y_i, z_i) \in P' : z_i \geq ax_i + by_i + c$$
$$\forall (x_j, y_j, z_j) \in Q' : z_j \leq ax_j + by_j + c$$

ראינו בשיעור שמספר הקודקודים של פוליטופ קמור לינארי במספר הפאות שלו, ולכן נקבל מספר לינארי של אילוצים (צריך גם להסביר מדוע פונק' המטרה לא חסומה אמ"מ P אינו מעל Q). לכן, נבנה ונפתור שתי בעיות LP בסיבוכיות זמן $O(m+n)$.

(b) נגדיר את $z_{\max}$ להיות ערך ה-z המקסימאלי שקודקוד של P או של Q משיג, ונגדיר את $z_{\min}$ בצורה סימטרית. בנוסף, נסמן $h = z_{\max} - z_{mix} + 1$.

נעלה את P כלפי מעלה במרחק h (זה מבטיח שהפוליטופים יהיו זרים) ונבצע את סעיף a על זוג הפוליטופים שהתקבלו. אם הערך שהוחזר הינו k, זאת אומרת שצריך להעלות את P כלפי מעלה במרחק $h-k$ על מנת ששני הפוליטופים ישיקו. נבצע פעולה סימטרית על מנת לבדוק מהו המרחק שיש להזיז את P כלפי מטה כך ששני הפוליטופים ישיקו, ונחזיר את הערך המינימאלי מבין שני הערכים שקיבלנו.