תרגול 2
אלגוריתמים חמדניים

אלגוריתמים 1 סמסטר א תשפ"ב

אלגוריתם חמדן

- **אלגוריתם חמדן**
אלגוריתם המוצא פתרון אופטימלי לבעיה, כך שבכל שלב נבחרת האפשרות שנראית כטובה ביותר באותו רגע.

- דוגמאות:
  - תשלום ע"י שימוש במינימום מטבעות
  - משחקי קלפים

- הערה:
אלגוריתם חמדן לא מוצא את הפתרון האופטימאלי עבור כל בעיה.

אלגוריתם חמדן מתאים לבעיות עם התכונות הבאות:

תכונת הבחירה החמדנית:

- בחירה **מקומית** אופטימלית (חמדנית) מובילה לפתרון אופטימלי **כולל**.

תת-מבנה אופטימלי:

- פתרון אופטימלי **לבעיה** מכיל בתוכו פתרונות אופטימליים **לתת- בעיות**.

בעית תזמון התדלוק

נתונות N מכוניות שממתינות לתדלוק.
עבור מכונית i נסמן את זמן התדלוק כך- Time i.

נרצה לקבוע את סדר התדלוק של המכוניות, כך שסך זמן ההמתנה של כל המכוניות יהיה מינימלי.

פתרון:

אלגוריתם אופטימלי שפותר את הבעיה- האלגוריתם החמדני.

- נמיין את המכוניות בסדר לא-יורד של זמני תדלוק:
  Time1 ≤ Time2 ≤ ... Time n
- נתדלק את המכוניות לפי הסדר, לאחר המיון.

זמן ריצה: θ (nlogn) - המיון לפי זמני תדלוק.

נכונות:

יש להוכיח את:

- תכונת הבחירה החמדנית
- תכונת תת המבנה האופטימלי.

תכונת הבחירה החמדנית

יש להוכיח ש**קיים** פתרון אופטימלי, בו **הבחירה הראשונה היא החמדנית**.

- נניח וקיים סדר אופטימלי של זמני תדלוק, בו המכונית הראשונה **אינה** בעלת זמן תדלוק מינימלי.
- נחליף את המכונית הראשונה עם המכונית בעלת זמן התדלוק המינימלי ← נקבל זמן המתנה **קצר יותר** בסך הכל.

תכונת תת-המבנה האופטימאלי

בהינתן פתרון אופטימלי לבעיה, נוכיח כי **אם** נוותר על המכונית הראשונה **אזי** נקבל פתרון אופטימלי לבעיה עבור N-1 מכוניות.

נניח כי בהינתן סדר אופטימלי למכוניות 1 עד N, סדר המכוניות 2 עד N אינו אופטימלי.

אזי **קיים** פתרון אחר שמשבץ את המכוניות 2 עד N בצורה אופטימלית.

נבחר את התזמון הבא:

- נשבץ את מכונית מספר 1, לאחר מכן נשבץ את המכוניות 2 עד N באופן האופטימלי.
- נקבל תזמון טוב יותר לקבוצת המכוניות 1 עד N.
- **סתירה** להנחה.

מסקנה:

בהינתן תזמון אופטימלי לבעיה כולה, נסיק כי גם התזמון לתת בעיה הוא אופטימלי.

בעיית מקדמי הפולינום

נתון מערך A של n מספרים טבעיים בלבד $A_0..A_{n-1}$, ומספר X גדול מ-1.

יש להציע סדר של האיברים במערך A,

כך שערך הפולינום $A_0+A_1X+A_2X^2+\ldots+A_{n-1}X^{n-1}$ יהיה **מקסימלי**.

- תכנן אלגוריתם חמדני שפותר את הבעיה.
- הוכח את נכונותו של האלגוריתם כולל הוכחת תכונת הבחירה החמדנית והוכחת תת מבנה אופטימלי.
- מהו זמן הריצה של האלגוריתם?

פתרון:

לצורך פתרון הבעיה, נמיין את המערך בסדר עולה.

תכונת הבחירה החמדנית:

בפתרון האופטימלי, A_0 הוא הערך הנמוך ביותר בתוך המערך A.

נניח שקיים פתרון אופטימלי אחר שבו באינדקס 0 במערך לא נמצא הערך הנמוך ביותר.

אזי הערך המינימלי A_{min} נמצא באינדקס, אחר במערך, גדול מ-0, נסמן אותו i.

נחליף בין A_{min} לבין A_0 ונקבל ערך פולינום שקטן ב- $(A_0 - A_{min})$ וגדל ב- $(A_0 - A_{min}) \cdot x^i$.

הואיל ו- X גדול מ-1, ערך הפולינום **גדל**.

תת המבנה האופטימלי:

אם קיים פתרון אופטימלי לפולינום כולו, הרי הפתרון בלי A_0 יהיה אופטימלי עבור הפולינום החל מ-X.

הוכחה: נניח בשלילה שהסדר של המערך כולו אופטימלי, אך הסדר הממוין של תת המערך A1 עד An-1 אינו אופטימלי עבור הפולינום $A_1x + \cdots + A_{n-1}x^{n-1}$.

קיים בהכרח סדר אופטימלי עבור תת המערך $A_1 \ldots A_{n-1}$ כך שנקבל ערך גבוה יותר לפולינום המתאים.

נוסיף את A_0 לפתרון האופטימלי עבור תת המערך ונקבל ערך **גבוה יותר** לפולינום כולו.

זאת בסתירה להנחה שהפתרון המקורי היה אופטימלי לפולינום כולו.

זמן ריצה: θ (nlogn) - מפעילים מיון של המקדמים.

מבצעים בקיוסק של ראובן

בקיוסק של ראובן החליטו על מבצע. כל מי שקונה זוג מוצרים מקבל את הזול מביניהם חינם. לאחר שבחרנו $2n$ מוצרים במחירים $p_1,\ldots,p_{2n}$, ברצוננו לסדר אותם בזוגות על מנת לקבל את ההנחה. הצע אלגוריתם המניב זוגות של מוצרים כך שהסכום הכולל שנאלץ לשלם על המוצרים הוא מינימאלי.

פתרון:

נמיין את המוצרים מהיקר לזול, לאחר מכן נבחר זוגות של מוצרים צמודים:

$(X_1\&X_2), (X_3\&X_4) \dots (X_{N-1}\&X_N)$

תכונת הבחירה החמדנית:

אם נשדך ל- X_1 ו- X_2 בני זוג אחרים, ארבעת המוצרים לפי חלוקת הזוגות החדשה תעלה יותר מארבעת המוצרים לפי חלוקת הזוגות המקורית
← נובע ישירות מהאופן שבו מיינו את המוצרים.

תכונת תת המבנה האופטימלי:

בהינתן פתרון אופטימלי ל-N זוגות של מוצרים, נניח בשלילה כי קיים פתרון אופטימלי אחר עבור N-1 זוגות מוצרים (לאחר שהורדנו "זוג" אחד של מוצרים) ונגיע לסתירה באותו האופן.

מספר מינימאלי של רציפים פתוחים בתחנת הרכבת

בהינתן לו"ז הגעות ויציאות של רכבות בתחנת רכבת, מצאו את מספר הרציפים המינימאלי הנדרש, על מנת למנוע עיכובים בכניסת הרכבות לתחנה.

נתון:

1) מערך A (Arrivals) ובו שעות ההגעה של רכבות הנכנסות לתחנה.
2) מערך D (Departures) ובו שעות היציאה של הרכבות מהתחנה.

פתרון:

1. נמיין את שני המערכים בסדר עולה.
2. נגדיר משתנה train_count שיספור לנו את כמות הרכבות בתחנה בכל רגע נתון.
3. נגדיר משתנה max_tc שישמור את המספר המקסימאלי של רכבות שהיו עד כה בו"ז בתחנה.
4. נעקוב בעזרת מצביעים על זמני ההגעה והיציאה של הרכבות עד שיעבור זמן ההגעה האחרון (while i < A.length).

a) אם הזמן הקרוב ביותר הוא של כניסת רכבת (A[i] < D[j]) , נוסיף 1 לסופר הרכבות (train_count++) ונעדכן את המספר המקסימאלי של רכבות בתחנה, במידה ועברנו את ערכו הקודם. לאחר מכן נקדם את המצביע של ההגעות (i++).

b) אם הזמן הקרוב הוא של יציאת רכבת (D[j] > A[i]), נחסיר 1 מסופר הרכבות (train_count--). לאחר מכן נקדם את המצביע של ההגעות (j++).

5. מקרה מיוחד: אם זמן ההגעה הקרוב = לזמן היציאה הקרוב, נדאג לשלח קודם את הרכבת היוצאת (train_count--) לפני שנקבל את הנכנסת.

max_tc = מספר הרכבות המקסימאלי בתחנה בכל רגע נתון = מספר הרצפים המינימאלי הנדרש

סיבוכיות: מיון = $O(n \cdot \log n)$ + מעבר פעם אחת על כל ההגעות והיציאות = $O(n)$. סה"כ:

$$\boldsymbol{O(n \cdot \log n)}$$

דוגמא:

$$\boldsymbol{A} = [\mathbf{2.00}, \mathbf{2.10}, \mathbf{5.00}, \mathbf{3.20}, \mathbf{3.50}, \mathbf{3.00}]$$

$$\boldsymbol{D} = [\mathbf{2.30}, \mathbf{3.55}, \mathbf{3.20}, \mathbf{4.30}, \mathbf{4.00}, \mathbf{5.20}]$$

לאחר מיון:

$$\boldsymbol{A} = [\mathbf{2.00}, \mathbf{2.10}, \mathbf{3.00}, \mathbf{3.20}, \mathbf{3.50}, \mathbf{5.00}]$$

$$\boldsymbol{D} = [\mathbf{2.30}, \mathbf{3.20}, \mathbf{3.55}, \mathbf{4.00}, \mathbf{4.30}, \mathbf{5.20}]$$

דוגמא:

$$\overset{i}{A} = [2.00, 2.10, 3.00, 3.20, 3.50, 5.00]$$

$$\overset{j}{D} = [2.30, 3.20, 3.55, 4.00, 4.30, 5.20]$$

רכבת מגיעה לרציף 1 בשעה 2.00 : train_count = 1, max_tc = 1

רכבת מגיעה לרציף 2 בשעה 2.10 : train_count = 2, max_tc = 2

רכבת יוצאת מרציף 1 בשעה 2.30 : train_count = 1, max_tc = 2

רכבת מגיעה לרציף 1 בשעה 3.00 : train_count = 2, max_tc = 2

רכבת יוצאת מרציף 2 בשעה 3.20 : train_count = 1, max_tc = 2

רכבת מגיעה לרציף 2 בשעה 3.20 : train_count = 2, max_tc = 2

רכבת מגיעה לרציף 3 בשעה 3.50 : train_count = 3, max_tc = 3

דוגמא:

לאחר מיון:

$$A = [2.00, 2.10, 3.00, 3.20, 3.50, \overset{i}{5.00}]$$

$$D = [2.30, 3.20, 3.55, \overset{j}{4.00}, 4.30, 5.20]$$

רכבת יוצאת מרציף 1 בשעה 3.55 : train_count = 2, max_tc = 3

רכבת יוצאת מרציף 2 בשעה 4.00 : train_count = 1, max_tc = 3

רכבת יוצאת מרציף 3 בשעה 4.30 : train_count = 0, max_tc = 3

רכבת מגיעה לרציף 1 בשעה 5.00 : train_count = 1, max_tc = 3

$i = A.length$. יצאנו מהלולאה. כעת נחזיר את ערכו של max_tc.

יוצא שהמספר המינימאלי של רציפים פתוחים הנדרשים לניהול הלו"ז

הנתון הינו : 3

בעיית השוטרים והגנבים

נתון מערך בגודל N שמכיל ערכים באופן הבא:

- כל תא במערך מכיל שוטר או גנב.
- כל שוטר יכול לתפוס גנב אחד בלבד.
- שוטר אינו יכול לתפוס גנב הנמצא במרחק של יותר מ- k תאים מהשוטר.

יש למצוא את המספר ה**מקסימלי** של גנבים, שאפשר לתפוס.

הצע אלגוריתם לפתרון הבעיה, הוכח את נכונותו וחשב את סיבוכיות זמן הריצה של האלגוריתם.

דוגמא:

P- שוטר, T- גנב.

$$input: \{P,T,T,P,T\}, k = 1$$

$$output: 2$$

$$input: \{P, T, P, P, T, T, P\}, k = 3$$

$$output: 3$$

פתרון:

1. מצא את האינדקס הנמוך ביותר שבו יש שוטר – p
   ואת האינדקס הנמוך ביותר שבו יש גנב – t.
2. אם $|p - t| \leq k$:
   "שדך" בין השוטר והגנב
   הגדל את p, t לאינדקסים הבאים של שוטר וגנב.
3. אחרת:
   הגדל את הקטן מבין p ו- t לאינדקס הבא
   חזור על 2 הצעדים האחרונים.
4. החזר את מספר ה-"שידוכים".


Any questions?