# פתרון תרגיל בית מספר 2

## שאלה 1

א. נשתמש באלגוריתם תכנות דינמי הידוע לפתרון בעית הסכומים החלקיים, אלא שהפעם יתכנו גם מחוברים שליליים, לכן לא נוכל להגביל את רוחב הטבלה ע"י הערך S. הרי אם קיימת קבוצה שמסתכמת ל-S, הסכומים של תתי-הקבוצות שלה יכולים להיות גם גדולים מ-S וגם שליליים. נסמן ב-P את סכום כל הערכים החיוביים ב-A, וב-N – סכום כל השליליים. כלומר, $P = \sum_{a_i>0} a_i$ ו-$N = \sum_{a_i<0} a_i$. אז הטבלה תכיל את העמודות עבור כל המספרים השלמים בין P ל-N (כולל).
האלגוריתם עצמו לא משתנה, יש רק לעדכן את תנאי השפה:

### איתחול ###
$P = \sum_{a_i>0} a_i\ ,\ N = \sum_{a_i<0} a_i\ ,\ \ t = table\ (P + |N| + 1) \times (n+1)$
for $j = N..P$ : $\quad t[0,j] = 0$
for $i = 0..n$ : $\quad t[i,0] = 1$
### חישוב הטבלה ###
for $i = 1..n$ :
    for $j = N..P$ :
        if $N \le j - a_i \le P$ : $\quad t[i,j] = t[i-1,j] \vee t[i-1, j-a_i]$
        else: $\quad t[i,j] = t[i-1,j]$
if $t[n,S] = 0$ : $\quad$ return $False$
### בניית קבוצת הסכום ע"ס הטבלה ###
$Q = \emptyset\ ,\ x = S$
for $i = n..1$ :
    if $t[i-1,x] = 0$ : $\quad Q = Q \cup \{a_i\},\ x = x - a_i$
return $True, Q$

ב. החלק הכבד ביותר באלגוריתם הוא מילוי הטבלה בגודל $(P + |N| + 1) \times (n+1)$. חישוב תא בודד הוא O(1), לכן סה"כ הסיבוכיות היא $O\left(\left(\sum_{i=1}^{n} |a_i|\right) \cdot n\right)$. זאת סיבוכיות פסאודו-פולינומית.

ג. <u>הטבלה</u>:

| | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 5 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

הערה: את השורה האחרונה אין צורך למלא את כולה, אלא רק תא $t[n,S]$, כי לא נצטרך ערכים אחרים בשורה הזאת. וספציפית בדוגמה הנ"ל קיבלנו 1 בעמודה 7 כבר בשורה 5, לכן יכולנו לא למלא את השורה 6 בכלל.

כעת, נפעיל את החלק האחרון של האלגוריתם ונקבל את הקבוצה הבאה (החישוב לפי התאים המסומנים): $Q = \{9, -3, 2, 4, -5\}$. (קיימים גם פתרונות אחרים)

## שאלה 2

<u>האלגוריתם:</u>

### הרצת האלג' להתאמת מחרוזות ###
$\pi, \tau = KMP(T, P)$
### חישוב מס' רישות בכל רישא של התבנית ###
$a[0] = 0$
for $i = 1..m:\quad a[i] = a[\pi[i]] + 1$
### חישוב מס' רישות של התבנית בסיפות של כל הרישות של הטקסט ###
for $i = 1..n:\quad A[i] = a[\tau[i]]$
return $A$

<u>נכונות:</u>

נסמן ב-$P_i$ את הרישא של P באורך $i$.
במערך $\pi$ המוחזר ע"י KMP, כל $\pi[i]$ מציין אורך הרישא הארוכה ביותר של P שקטנה מ-$i$ ונכנסת כסיפא בתוך $P_i$. לכן מס' רישות לא ריקות של P שהן סיפות של $P_i$ שווה למס' רישות כאלה ברישא באורך $\pi[i]$ פלוס אחד עבור $i$ עצמו. לכן בתום תהליך החישוב של המערך $a$, כל $a[i]$ מכיל את מס' רישות לא ריקות שהן סיפות של $P_i$.
מאחר ו-$\tau[i]$ הוא אורך הרישא הארוכה ביותר של P שהיא גם סיפא של $T[1..i]$, אז $a[\tau[i]]$ זהו בדיוק מס' רישות לא ריקות של P שהן סיפות של $T[1..i]$.

<u>סיבוכיות:</u>

- סיבוכיות KMP היא $O(m+n)$.
- חישוב $a$ הוא $O(m)$.
- חישוב $A$ הוא $O(n)$.
- סה"כ $O(m+n)$.

## שאלה 3

א.

<u>טבלת המצבים:</u>

| מצבים \ תווים | A | B | C |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 1 | 2 | 1 | 0 |
| 2 | 0 | 3 | 0 |
| 3 | 2 | 4 | 0 |
| 4 | 5 | 1 | 0 |
| 5 | 0 | 6 | 0 |
| 6 | 2 | 4 | 7 |
| 7 | 8 | 1 | 0 |
| 8 | 0 | 9 | 0 |
| 9 | 2 | 1 | 0 |

ב. הרצת האוטומט על T:

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | A | B | A | B | A | B | A | B | B | A | B | C | A | A |
| state | 0 | 1 | 2 | 3 | 2 | 3 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 0 |

לא יודפסו הודעות, כי התבנית לא נמצאת בתוך הטקסט.

ג.

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $\pi$ | 0 | 0 | 1 | 1 | 2 | 3 | 0 | 0 | 1 |

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\tau$ | 0 | 1 | 2 | 3 | 2 | 3 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 0 |

## שאלה 4

### PROGRAM 1

א. סיבוכיות $O(n)$.
הסבר: בלולאה, $i$ רץ מ-1 עד $n$ ומקודם ב-1 בכל איטרציה, לכן יהיו $n$ איטרציות. וכל איטרציה היא $O(1)$.

ב. התוכנית עלולה לא למצוא אף מופע, למרות שיש.
הסבר: לדוגמה, אם T = "ARARAT" ו- P = "RAT", התוכנית לא תמצא את P בתוך T, כי כשנסיון זיהוי נכשל באמצע, התכנית לא חוזרת אל התווים שכבר זוהו.

### PROGRAM 2

א. סיבוכיות $O(n \cdot m)$.
הסבר: במקרה של זיהוי תבנית בתוך טקסט, $i$ מוחזר $m-1$ תווים אחורה.
לדוגמה, אם $T = "\underbrace{AAA \dots A}_{n}"$ ו- $P = "\underbrace{AAA \dots A}_{m}"$, התוכנית תרוץ $(n-m) \cdot m$ איטרציות.

ב. התוכנית עלולה לא למצוא אף מופע, למרות שיש.
הסבר: לדוגמה, אם T = "ARARAT" ו- P = "RAT", התוכנית לא תמצא את P בתוך T, כי כשנסיון זיהוי נכשל באמצע, התכנית לא חוזרת אל התווים שכבר זוהו (כמו תכנית 1).

### PROGRAM 3

א. סיבוכיות $O(n \cdot m)$.
הסבר: בלולאה, $k$ רץ מ-1 עד $n-m+1$ ולכל ערך של $k$ מתבצע נסיון זיהוי (כושל או מוצלח) באורך עד $m$ תווים. לדוגמה, אם $T = "\underbrace{AAA \dots A}_{n}"$ ו- $P = "\underbrace{AAA \dots A}_{m}"$, התוכנית תרוץ $(n-m) \cdot m$ איטרציות.

ב. התוכנית מוצאת את כל המופעים.
הסבר: התוכנית, למעשה, מממשת את האלגוריתם הנאיבי שפשוט עובר על כל האפשרויות.