מבוא למדעי המחשב מ'/ח' (234114 \ 234117)

סמסטר חורף תשפ"א

מבחן מסכם מועד ב', 18 במרץ 2021

מספר סטודנט: | | | | | | | | | | *רשום/ה לקורס:* | | | | | | 2 | 3 | 4 | 1 | 1 |

משך המבחן: 3 שעות.

חומר עזר: אין להשתמש בכל חומר עזר.

הנחיות כלליות:

- מלאו את הפרטים בראש דף זה ובדף השער המצורף, בעט בלבד.
- בדקו שיש 22 עמודים (4 שאלות) במבחן, כולל עמוד זה.
- כתבו את התשובות על טופס המבחן בלבד, במקומות המיועדים לכך. שימו לב שהמקום המיועד לתשובה אינו מעיד בהכרח על אורך התשובה הנכונה.
- העמודים הזוגיים בבחינה ריקים. ניתן להשתמש בהם כדפי טיוטה וכן לכתיבת תשובותיכם. סמנו טיוטות באופן ברור על מנת שהן לא תיבדקנה.
- יש לכתוב באופן ברור, נקי ומסודר, **ובעט בלבד**.
- בכל השאלות, הנכם רשאים להגדיר ולממש פונקציות עזר כרצונכם. לנוחיותכם, אין חשיבות לסדר מימוש הפונקציות בשאלה, ובפרט ניתן לממש פונקציה לאחר השימוש בה.
- אלא אם כן נאמר אחרת בשאלות, **אין להשתמש בפונקציות ספריה או בפונקציות שמומשו בכיתה**, למעט פונקציות קלט/פלט והקצאת זיכרון (`malloc, free`). ניתן להשתמש בטיפוס `bool` המוגדר ב-`stdbool.h`
- אין להשתמש במשתנים סטטיים וגלובאליים אלא אם נדרשתם לכך מפורשות.
- כשאתם נדרשים לכתוב קוד באילוצי סיבוכיות זמן/מקום נתונים, אם לא תעמדו באילוצים אלה תוכלו לקבל בחזרה מקצת הנקודות אם תחשבו נכון ותציינו את הסיבוכיות שהצלחתם להשיג.
- נוהל "לא יודע": אם תכתבו בצורה ברורה "לא יודע/ת" על שאלה (או סעיף) שבה אתם נדרשים לקודד, תקבלו 20% מהניקוד. דבר זה מומלץ אם אתם יודעים שאתם לא יודעים את התשובה.
- נוסחאות שימושיות:

$$1+\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{n}=\Theta(\log\ n) \qquad \log 1+\log 2+\ldots+\log n=\Theta(n\ \log\ n)$$

$$1^k+2^k+3^k+\ldots+n^k=\Theta(n^{k+1}) \qquad 1+k+k^2+k^3+\ldots+k^n=\Theta(k^n),\ k>1$$

צוות הקורס 234114/7

מרצים: פרופ' תומר שלומי (מרצה אחראי), פרופ' ניר אילון, מר איהאב וותד, גב' יעל ארז **מתרגלים:** גיא בר שלום, מג'ד חורי, עמית ברכה, קטרין חדאד, רואי בנימין, תומר לנגה, מתן ממיסטבלוב, יבגני נוגין, ירון חי, תום הרמן, עדי גרוס, הראל וקנין, סמיר מסעד, דמיטרי רבינוביץ' (מתרגל אחראי).

בהצלחה!

שאלה 1 (25 נקודות):

א. (8 נקודות) חשבו את סיבוכיות הזמן והמקום של הפונקציה f1 המוגדרת בקטע הקוד הבא, כפונקציה של n. אין צורך לפרט שיקוליכם. חובה לפשט את הביטוי ככל שניתן.

```
int aux(int m, int n)
{
    if (n <= 1)
        return 1;

    return m * aux(m, n / 2);
}

void f1(int n)
{
    for (int i = aux(n, n); i > 0; i /= 2)
        printf(" * ");
}
```

סיבוכיות זמן: $\Theta($_____$)$ סיבוכיות מקום: $\Theta($_____$)$

ב. (9 נקודות): חשבו את סיבוכיות הזמן והמקום של הפונקציה f2 המוגדרת בקטע הקוד הבא, כפונקציה של n. אין צורך לפרט שיקוליכם. חובה לפשט את הביטוי ככל שניתן.

```
void f2(int n)
{
    int x = 0;
    for (int i = 1; i < n; i *= 2)
        for (int j = 1; j < n; j *= 2)
            x++;
}
```

סיבוכיות זמן: $\Theta($_____$)$ סיבוכיות מקום: $\Theta($_____$)$

ג. (8 נקודות): חשבו את סיבוכיות הזמן והמקום של הפונקציה f3 המוגדרת בקטע הקוד הבא, כפונקציה של n. אין צורך לפרט שיקוליכם. הניחו שסיבוכיות הזמן של malloc(n) היא $\Theta(1)$, וסיבוכיות המקום של malloc(n) היא $\Theta(n)$. חובה לפשט את הביטוי ככל שניתן.

```
int* f3(int n)
{
    if (n <= 1) return NULL;

    for (int j = 0; j < n; ++j)
        printf("-");

    int *ptrs[4] = { NULL };
    for (int i = 0; i < 4; ++i)
        ptrs[i] = f3(n / 2);

    for (int i = 0; i < 4; ++i)
        free(ptrs[i]);

    return malloc(n * n);
}
```

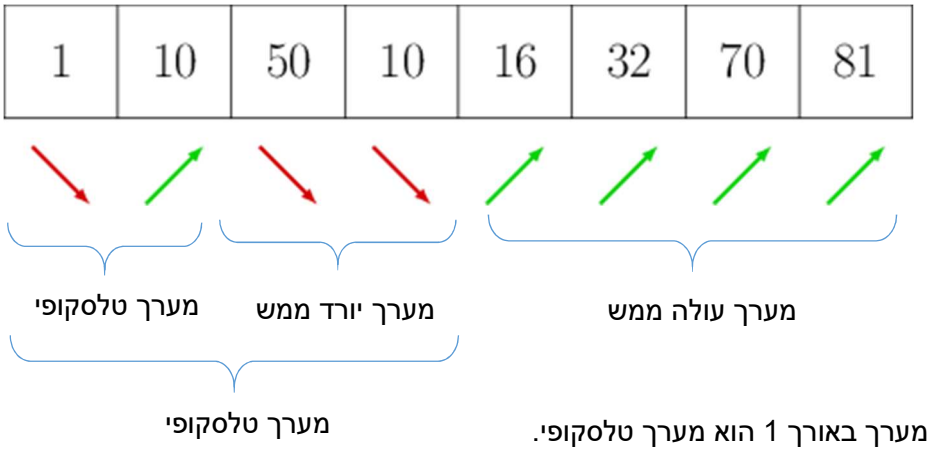
סיבוכיות זמן: $\Theta($_____$)$ סיבוכיות מקום: $\Theta($_____$)$

שאלה 2 (25 נק')

בשאלה זו נניח כי אורך המערך הוא חזקה שלמה של 2, ויש בו לפחות 2 איברים.

נגדיר מערך של מספרים שלמים באורך n כ***מערך טלסקופי***, אם החצי הימני של המערך ממוין בסדר **עולה ממש או יורד ממש** והחצי השמאלי שלו הוא *מערך טלסקופי*.

לדוגמה: המערך הבא הוא ***מערך טלסקופי***:



הערה: מערך באורך 1 הוא מערך טלסקופי.

ממשו פונקציה המקבלת מערך של מספרים שלמים `arr` שהוא מערך טלסקופי ואת הגודל שלו $n = 2^k$ שחתימתה:

```
int FindInTelescope(int arr[], int n, int x);
```

הפונקציה מחזירה אינדקס לתא במערך, בו נמצא המספר x. אם לא קיים אינדקס כזה, הפונקציה תחזיר -1. אם קיימים כמה אינדקסים כאלה הפונקציה תחזיר אחד מהם.

לדוגמה: עבור המערך לעיל ו- $x = 10$, ערכי החזרה האפשריים הם 1 או 3. עבור $x = 15$ הפונקציה תחזיר -1.

דרישות:

- ממשו את הפונקציה בסיבוכיות זמן $O((\log n)^2)$, כאשר n הוא אורך המערך.
- אם לא עמדתם בדרישות הסיבוכיות, אנא ציינו כאן את הסיבוכיות שהגעתם אליה:

זמן _____ מקום נוסף _____

`int FindInTelescope (int arr[], int n, int x)`
`{`

שאלה 3 (25 נקודות) :

עליכם לממש את הפונקציה שחתימתה:

```
int FindMaxSlimSubstring(char* str1, char* str2);
```

הפונקציה מקבלת שתי מחרוזות המורכבות מאותיות **קטנות באנגלית** בלבד ומחזירה את האורך של התת-מחרוזת הארוכה ביותר של str1 שלא מכילה **כלל** אותיות המופיעות במחרוזת str2. שימו לב, אורך מחרוזת ריקה הוא 0.

דוגמאות:

```
char str1[] = "hello", char str2[] = "low";
FindMaxSlimSubstring (str1, str2); // returns 2
// since "he" is the longest sub-string that does not contain
even a single letter from str2

char str1[] = "hello", char str2[] = "hole";
FindMaxSlimSubstring (str1, str2); // returns 0
// since every single non-empty sub-string of str1 contains at
least one letter of str2

char str1[] = "curious", char str2[] = "coast";
FindMaxSlimSubstring (str1, str2); // returns 3
// since "uri" is the longest substring that does not contain
letters that appear in str2
```

הערות:

- ניתן להניח כי המילים מורכבות מאותיות אנגליות קטנות בלבד.

דרישות:

- סיבוכיות זמן $O(n_1 + n_2)$, כאשר n_1 הינו אורך של המחרוזת str1, ו-n_2 הינו אורך של המחרוזת str2 וסיבוכיות מקום $O(1)$.

אם לפי חישוביכם לא עמדתם בדרישות הסיבוכיות אנא ציינו כאן את הסיבוכיות שהגעתם אליה:

זמן _____ מקום נוסף _____

`int FindMaxSlimSubstring(char* str1, char* str2)`
`{`

שאלה 4 (25 נקודות) :

סוכן הפצה בחברת COWEED מתכנן את מסלול הנסיעה **הקצר ביותר** בין **כל** נקודות היישובים הפרוסים ברחבי הארץ. בעקבות מגפת COVID-19 דרש משרד הבריאות, כי המסלול **לא יעבור ביותר מ-k** יישובים עוקבים המסומנים בצבע אדום ברמזור הצבעים של המשרד.

ממשו פונקציה

```
int shortest_path(int dist [N][N], bool red_ones[N], int k);
```

שמקבלת את מפת המרחקים בין היישובים `dist`, כאשר המרחק מיישוב `i` ליישוב `j` נתון ב-`dist[i][j]` (יתכן ואין כביש ישיר בין היישובים, במקרה זה `dist[i][j]=-1`), ואת סימון הרמזור של משרד הבריאות `red_ones`, כאשר העובדה שיישוב `i` הוא אדום תסומן ב-`true` במקום `i`. בנוסף תקבל הפונקציה פרמטר אי שלילי `k`.

שימו לב,

- הפונקציה תחזיר את האורך של מסלול הקצר ביותר העובר בכל היישובים והעומד בקריטריונים, אחרת תחזיר `-1`.
- טבלת המרחקים **אינה** בהכרח סימטרית.
- ניתן להניח שבכל איברי האלכסון רשום `0`.
- ניתן לעבור ביישוב פעם אחת בדיוק, והמסלול לא חייב להתחיל ביישוב שמספרו `0`.
- ניתן להניח כי `N` מוגדר ב `define` והוא חיובי.

דוגמה: בהינתן טבלת המרחקים הבאה, סימון רמזור אדום עבור יישובים `1` ו-`2`, `k=1`, המסלול הקצר ביותר הוא מסלול `2 - 3 - 1 - 0`, באורך `4`.

source \ destination	0	1	2	3
0	0	1	2	3
1	1	0	1	2
2	3	1	0	1
3	2	-1	1	0

אומנם קיים מסלול קצר יותר 3 – 2 – 1 – 0, באורך 3, אך הוא עובר בשני יישובים אדומים עוקבים (1 ו-2) ולכן אינו חוקי.
עבור אותה טבלת מרחקים, אותו סימון רמזור אדום, k=0, הפונקציה תחזיר 1- כיוון שלא קיים מסלול העובר רק דרך יישובים הלא אדומים.

הערות:

- יש להשתמש בשיטת backtracking כפי שנלמדה בכיתה.
- בשאלה זו אין דרישות סיבוכיות.
- ניתן ומומלץ להשתמש בפונקציות עזר (ויש לממש את כולן).

```
int shortest_path(int dist [N][N], bool red_ones[N], int k)
{
```