



מבוא למדעי מחשב מ' / ח' (234114 / 234117)

סמסטר חורף תשע"ג

מבחן מסכם מועד א', 3 מרץ 2013

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
שם פרטי	שם משפחה	מספר סטודנט							

משך המבחן: 2.5 שעות.
חומר עזר: אין להשתמש בכל חומר עזר (מודפס או ממוחשב).

הנחיות כלליות:

- מלאו את הפרטים בראש דף זה ובדף השער המצורף, בעט בלבד.
- בדקו שיש 18 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 לכתוב תשובותיכם. סמנו טיוטות באופן ברור על מנת שהן לא תיבדקנה.
- יש לכתוב באופן ברור, נקי ומסודר. ניתן בהחלט להשתמש בעיפרון ומחקק, פרט לדף השער אותו יש למלא בעט.
- בכל השאלות, הינכם רשאים להגדיר ולממש פונקציות עזר כרצונכם. לנוחיותכם, אין חשיבות לסדר מימוש הפונקציות בשאלה, ובפרט ניתן לממש פונקציה לאחר השימוש בה.

הנחיות תכנות כלליות, אלא אם מצוין אחרת בשאלה:

- אין להשתמש בפונקציות ספרייה או בפונקציות שמומשו בכיתה, למעט פונקציות קלט/פלט והקצאת זיכרון (`malloc`), אלא אם נכתב אחרת.
- אין להשתמש במשתנים סטטיים וגלובאליים אלא אם נדרשתם לכך מפורשות.
- ניתן להשתמש בהקצאות זיכרון בסגנון C99 (מערכים בגודל משתנה), בכפוף לדרישות סיבוכיות זיכרון.
- ניתן להשתמש בטיפוס `bool` המוגדר ב-`stdbool.h`.

צוות הקורס 234114/7

מרצים: ד"ר תמיר לוי, ד"ר רן רובינשטיין, פרופ' רון קימל (מרצה אחראי).

מתרגלים: תהילה מייזלס, שי גרץ, סינטיה דיזנפלד, נחשון כהן, נדיה לבאי, ורד כהן, תומר ארזי, יעל ארז, אסף ישראל (מתרגל אחראי).

בהצלחה!



- 2 -



שאלה 1 (20 נקודות)

בכל אחד מקטעי הקוד הבאים, חשבו את סיבוכיות הזמן והמקום של הפונקציה `func` כתלות ב-`n`:

א.

```
int aux(int n, int x)
{
    while (n > x) {
        x *= x;
    }
    return x;
}

int func(int n)
{
    return aux(n,2);
}
```

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

ב.

```
void aux(int n)
{
    int m=0;
    for (int i=0; i<n; ++i) {
        m += i;
    }
    printf("%d\n", m);
}

int func(int n) {
    if (n < 2) {
        return 0;
    }
    aux(func(n/2));
    return n;
}
```

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



- 4 -



ג.

```
int aux(int n)
{
    int m=0, i=n;
    while (i>0) {
        m += i;
        i /= 2;
    }
    return m*n;
}

int func(int n)
{
    int sum=0;
    for (int i=0; i<n; ++i)
    {
        int k=aux(n);
        while (k>0) {
            sum++;
            k--;
        }
    }
    return sum;
}
```

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



- 6 -



שאלה 2 (20 נקודות)

בשאלה זו נתון מערך ממזין של מספרים חיובים, שלמים השונים זה מזה. ברצוננו להגדיל את ערכו של אחד מאברי המערך $a[i]$ ב- x , תוך שמירת המערך ממזין. לדוגמה, עבור המערך a הבא באורך 7,

1	2	4	5	7	8	9
---	---	---	---	---	---	---

הגדלת האיבר $a[1]$ ב-4 תיתן את התוצאה הבאה (לאחר סידור מחדש של המערך):

1	4	5	6	7	8	9
---	---	---	---	---	---	---

ממשו את הפונקציה `update_arr` למטה, המקבלת את המערך הממזין a , את אורכו n , אינדקס i של איבר כלשהו במערך, וערך x שבו יש להגדיל את $a[i]$, ומבצעת את הפעולה הנדרשת תוך שמירת a ממזין.

הערות:

- על הפונקציה לעבוד בסיבוכיות זמן $O(x)$ וסיבוכיות מקום $O(1)$.
- הסבירו בקצרה במקום המתאים מדוע הפתרון שלכם עומד בזמן הריצה הדרוש. כדאי להשתמש בעובדה שכל איברי מערך הקלט שונים זה מזה.

הסבר לסיבוכיות הזמן:

```
void update_arr(int a[], int n, int i, unsigned int x) {
```



- 8 -



שאלה 3 (30 נקודות)

כזכור, בבעיית מגדלי Hanoi אנו מתבקשים להעביר n דיסקיות בגדלים שונים מעמודה $from$ לעמודה to ע"י שימוש בעמודה שלישית via . התנאי הוא שבשום שלב לא תונח דיסקית גדולה על דיסקית קטנה ממנה.

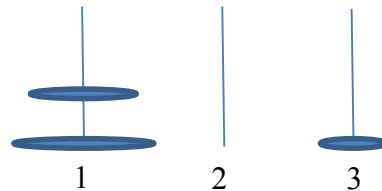
בכיתה ראינו פיתרון רקורסיבי הפותר את הבעיה עבור n דיסקיות במספר צעדים $2^n - 1$ בדיוק.

בשאלה זו נרצה לממש פונקציה המחשבת את המיקום (מספר העמודה) של כל דיסקית בבעיית מגדלי Hanoi לאחר k צעדים של האלגוריתם. הפונקציה תקבל את מספר הדיסקיות n , את עמודות המקור והיעד $from$ ו- to , את מספר צעדי האלגוריתם k , וכן מערך פלט $a[]$ באורך n . על הפונקציה לכתוב לתא $a[i]$ את מספר העמודה בה נמצאת הדיסקית ה- i לאחר k צעדים של האלגוריתם. שימו לב ש- $a[0]$ צריכה להכיל את העמודה של הדיסקית הגדולה ביותר, ו- $a[n-1]$ את העמודה של הדיסקית הקטנה ביותר. חתימת הפונקציה:

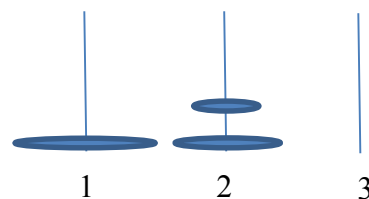
```
void hanoiState(int n, int from, int to, int k, int a[]);
```

דוגמאות:

```
hanoiState(3,1,3,1,a);  
a = [1,1,3]
```



```
hanoiState(3,1,3,3,a);  
a = [1,2,2]
```



הערות:

- על הפתרון להיות רקורסיבי ולעמוד בסיבוכיות זמן $O(n)$ וסיבוכיות מקום $O(n)$, כאשר n הוא מספר הדיסקיות.
- לצורך הפתרון ניתן להשתמש בפונקציות עזר מתמטיות ולהניח שהן פועלות בסיבוכיות זמן ומקום $O(1)$. אין צורך לממש פונקציות אלו, אך יש להסביר בקצרה את פעולתן.



- 10 -

הטכניון, מכון טכנולוגי לישראל מבוא למדעי המחשב מ"ח'

```
void hanoiState(int n, int from, int to, int k, int a[]) {
```



- 12 -



שאלה 4 (30 נקודות)

לחברת משלוחים מגיעים מדי יום n פריטים שצריכים להישלח אל למעבר לים. לכל פריט שמתקבל יש משקל כלשהו הנתון במערך $w[]$, כאשר $w[i]$ הוא משקלו של הפריט ה- i .

בבעלות החברה m מכולות זהות, היכולות להכיל כל אחת משקל c לכל היותר. שימו לב שניתן לארוז יותר מפריט אחד במכולה, אך לא ניתן שסכום משקלי הפריטים במכולה יחרוג מ- c .

בשאלה זו נרצה לממש פונקציה בשם `pack` המקבלת את רשימת הפריטים למשלוח, ומחזירה `true` אם ניתן לארוז את כל הפריטים ב- m מכולות, ו-`false` אחרת. חתימת הפונקציה:

```
bool pack(int w[], int n, int m, int c);
```

לדוגמה:

```
int w[7] = {1,3,2,1,1,1,2};  
pack(w, 7, 3, 4);
```

במקרה זה הפונקציה תחזיר `true` כיוון שניתן לארוז את שבעת הפריטים בשלוש מכולות בעלות קיבולת 4 כל אחת (למשל, ארבעה פריטים במשקל 1 במכולה ראשונה, שני פריטים במשקל 2 במכולה שנייה, ופריט אחד במשקל 3 במכולה שלישית).

הערות:

- ניתן להניח כי הקיבולת של כל מכולה וכן משקלי כל הפריטים חיוביים ממש.
- על הפתרון להיות רקורסיבי ולהשתמש ב-backtracking.
- ניתן להגדיר פונקציות עזר כרצונכם.



- 14 -





- 16 -



- 17 -



- 18 -