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

סמסטר אביב 2009

מבחן מסכם מועד ב', 25 ספטמבר 2009

מספר סטודנט

234114 / 234117

מספר קורס (הקף בעיגול)

משך המבחן: 115 דקות
חומר עזר: אין להשתמש בכל חומר עזר בכתב, מודפס או אלקטרוני.

הנחיות והוראות:

- מלאו את הפרטים בראש דף זה ובדף השער.
- בדקו שיש 14 עמודים (3 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. **ניתן בהחלט להשתמש בעיפרון ומחק (למרות מה שכתוב בדף הראשון!).**
- אין לכתוב הערות והסברים לתשובות אם לא נתבקשתם מפורשות לכך.
- בכל השאלות, הינכם רשאים להגדיר (ולממש) פונקציות עזר כרצונכם.
- **אין להשתמש בפונקציות ספריה, או בפונקציות שמומשו בכיתה אלא אם צוין אחרת במפורש בשאלה, למעט פונקציות קלט פלט והקצאת זיכרון (malloc).**
- בכל שאלה ניתן להשתמש בפונקציות המוגדרות בסעיפים קודמים של אותה שאלה גם אם לא פתרתם סעיפים אלו.

שאלה	ערך	הישג	בודק
1	34		
2	33		
3	33		
סה"כ	100		

צוות הקורס 234114/7

מרצים: פרופ' חבר יובל רבני (מרצה אחראי), מר רן רובינשטיין.

מתרגלים: זהר קרנין (מתרגל אחראי)
דן רביב, אייל רגב, אופיר ווברר, אייל רוזנברג, רועי אדדי.

בהצלחה!

שאלה 1 (34 נקודות)

בשאלה הבאה נתייחס לקטע הקוד הבא:

```
int *what(int *le);

int main() {
     int i,n, *a = what(&n);
     printf("%d ", n);
     for (i=0; i<n; i++)
           printf("%d ", a[i]);
     return 0;
}

int *what(int *le) {
    int x;
    int sp=1, si=0, i;
    int *a = (int *)malloc(sp * sizeof(int));
    while ( scanf(" %d", &x) == 1 ) {
        if (si >= sp) {
            int *t = (int *)malloc(2*sp*sizeof(int));
            for (i=0; i<sp; i++) {
                t[i] = a[i];
            }
            free(a);
            a=t;
            sp *= 2;
        }

        a[si] = x;
        for (i = si; i>0; i--) {
            if ( a[i] < a[i-1] ) {
                int t = a[i];
                a[i] = a[i-1];
                a[i-1]= t;
            }
            else
                break;
        }
        si++;
    }
    *le = si;
    return a;
}
```

חלק א' (18 נקודות)

מה תדפיס התוכנית בהינתן הקלט הבא (מהמקלדת)? (9 נקודות)

```
3 2 r
```

```
2 2 3
```

מה תדפיס התוכנית בהינתן הקלט הבא (מהמקלדת)? (9 נקודות)

```
24  -4   55  33  22  3 -55  a
```

```
7 -55 -4 3 22 24 33 55
```

חלק ב' (16 נקודות)

מה סיבוכיות הזמן והמקום של התוכנית בתלות בערך שקיבל המשתנה n? נמקו בקצרה. (10 נקודות)

סיבוכיות זמן: Θ(_____) סיבוכיות מקום נוסף: Θ(_____)

התוכנית מקצה מקום עבור מערך בן לפחות n איברים המכיל את המספרים שהוזנו בסדר ממויין לא יורד. סיבוכיות הזמן היא n^2 שכן הזמן הנדרש עבור הכנסת המספר ה-i הוא i. לכן, הזמן הכולל הוא

$1+2+3+4+\ldots+n = \Theta(n^2)$

סיבוכיות המקום הנוסף היא n שכן המערך בו נמצאים המספרים הינו בגודל n עד 2n.

איך (אם בכלל) תשתנה התשובה לסיבוכיות **המקום** אם נוריד את השורה free(a)? נמקו בקצרה. (6 נקודות)

סיבוכיות מקום נוסף: Θ(_____)

אם נוריד את השורה, אנו נקצה בסך הכל כמות זיכרון של

$1+2+4+8+\ldots+n= \Theta(n)$

זהו טור הנדסי שסכומו הוא Θ(n). לכן, סיבוכיות המקום נותרת Θ(n).

שאלה 2 (33 נקודות)

מטריצה בגודל NxN נקראת מטריצת *אדמר* כאשר כל איבריה הם 1,-1+ ובין כל 2 שורות שלה יש בדיוק N/2 ערכים שונים.
עבור ערך N שהוא חזקה של 2, ניתן לבנות מטריצות *אדמר* באופן הבא. המטריצות הראשונות הן:

$$\mathbf{H}(1) = (1), H(2) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, H(4) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

באופן כללי, H(2N) מתקבלת מ 4 עותקים של H(N), באופן הבא:

$$\mathbf{H}(2N) = \begin{pmatrix} H(N) & H(N) \\ H(N) & -H(N) \end{pmatrix}$$

כאשר הכוונה ב H(N)- הוא להיפוך כל הסימנים במטריצה H(N).

כתבו פונקציה רקורסיבית <u>ללא לולאות</u> שמקבלת מטריצה בגודל NxN (N מוגדר ב-#define) וממלאת אותה בערכים 1, 1-, כך שתהיה מטריצת אדמר. ניתן להניח כי N הוא חזקה של 2.
חתימת הפונקציה:

```
void fill_hadamard(int a[N][N])
```

רמז: מומלץ להשתמש בפונקציית עזר רקורסיבית עם החתימה.

```
void fill_hadamard_aux(int a[N][N], int __, int __, int __, int __);
```

מה סיבוכיות הזמן והמקום של הפונקציה שכתבתם בתלות ב-N? נמקו בקצרה.

סיבוכיות זמן: Θ(_____)	סיבוכיות מקום נוסף: Θ(_____)

```
void fill_hadamard(int a[N][N]) {
  fill_hadamard_aux(a,0,0,N,1);
}

void fill_hadamard_aux(int a[N][N], int top, int left, int
size, int sign) {
  if (size == 1) {
    a[top][left] = sign;
    return;
  }
  fill_hadamard_aux(a, top, left, size/2, sign);
  fill_hadamard_aux(a, top+size/2, left, size/2, sign);
  fill_hadamard_aux(a, top, left+size/2, size/2, sign);
  fill_hadamard_aux(a, top+size/2, left+size/2, size/2,
    -sign);
}
```

שאלה 3 (33 נקודות)

חלק א' (17 נקודות)

כתבו פונקציה המקבלת מערך ממויין a של מספרים שלמים, את גודלו n, ומספר נוסף x, ומחזירה את האינדקס האחרון בו מופיע המספר x. במידה והמספר אינו מופיע במערך, יש להחזיר את הערך 1-.

דוגמא: עבור המערך
{-5,-5,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3,67,67,99,150}
והמספר x = -5, הפונקציה תחזיר את הערך 1.
עבור אותו המערך והמספר x = 1, הפונקציה תחזיר את הערך 9.
עבור אותו המערך והמספר x=8, הפונקציה תחזיר את הערך 1-.
על הפונקציה לעבוד בסיבוכיות זמן O(log n)

```
int find_last(int a[], int n, int x) {
  int high = n, low = 0, mid;
  while (low < high) {
    mid = (low+high)/2;
    if (a[mid]==x && (mid==n-1 || a[mid+1] > x) )
       return mid;
    if (a[mid]<=x) {
      low = mid+1;
    } else {
      high = mid;
    }
  }
  return -1;
}
```

חלק ב' (16 נקודות)

כתבו פונקציה המקבלת מערך ממויין a של מספרים שלמים, את גודלו n, ומספר נוסף x. הפונקציה מחזירה את מספר המופעים של המספר x במערך a.

דוגמא: עבור המערך
{-5,-5,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3,67,67,99,150}
והמספר x = -5, הפונקציה תחזיר את הערך 2.
עבור אותו המערך והמספר x = 2, הפונקציה תחזיר את הערך 7.
עבור אותו המערך והמספר x=8, הפונקציה תחזיר את הערך 0.
על הפונקציה לעבוד בסיבוכיות זמן O(log n)

```
int find_first(int a[], int n, int x);

int count (int a[], int n, int x) {
  int last = find_last(a,n,x);
  if (last == -1) return 0;
  return last-find_first(a,n,x)+1;
}

int find_first(int a[], int n, int x) {
  int high = n, low = 0, mid;
  while (low < high) {
    mid = (low+high)/2;
    if (a[mid]==x && (mid==0 || a[mid-1] < x) ) return mid;
    if (a[mid]<x) {
      low = mid+1;
    } else {
      high = mid;
    }
  }
  return -1;
}
```