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

סמסטר חורף תשס"ו

פתרון מבחן מסכם מועד א', 15 פברואר 2006

מספר סטודנט	*שם משפחה*	*שם פרטי*

משך המבחן: 3 שעות.
חומר עזר: אין להשתמש בכל חומר עזר בכתב, מודפס או אלקטרוני.

הנחיות והוראות:

- מלאו את הפרטים בראש דף זה.
- בדקו שיש 9 עמודים (5 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.
- 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 בשאלה.**

שאלה	ערך	הישג	בודק
1	20		
2	20		
3	20		
4	20		
5	20		
סה"כ	100		

צוות הקורס 234114
מרצים: סאהר אסמיר, פרופ' רון קימל (מרצה אחראי).
מתרגלים: עידן בן-הרוש, גיא סלע, ולנטין קרבצוב, מרק גינזבורג, רן רובינשטיין (מתרגל אחראי).

צוות הקורס 234117
מרצים: ארז חדד, ויטלי סקצ'ק, פרופ' רון קימל (מרצה אחראי).
מתרגלים: שיאון שחורי, ארקדי פיורו, רג'א ג'יריס, אסנת טל, מרק גינזבורג, רן רובינשטיין (מתרגל אחראי).

בהצלחה!

שאלה 1 (20 נקודות)

סעיף א (10 נקודות)

בשאלה זו נתייחס לאותיות האנגליות הקטנות a…z כספרות, כאשר 'a' ערכה 0, 'b' ערכה 1, 'c' ערכה 2 וכן הלאה עד 'z' שערכה 25. כעת, אותיות אלו ישמשו אותנו לצורך ייצוג מספרים ממשיים חיוביים (גדולים או שווים ל-0) בבסיס 26. כל מספר ייוצג כמחרוזת בפורמט הבא:

$$a_{k-1}a_{k-2}...a_1a_0 \,.\, a_{-1}a_{-2}...a_{-p}$$

כאשר כל a_i מייצג אות אנגלית קטנה. במספר זה ישנם k "ספרות" לפני הנקודה ועוד p "ספרות" אחרי הנקודה, ולצורך השאלה ניתן להניח כי k≥1 וגם p≥1. ערכו של המספר בייצוג זה נתון על ידי הביטוי:

$$a_{k-1}\cdot 26^{k-1} + a_{k-2}\cdot 26^{k-2} + ... + a_0\cdot 26^0 + a_{-1}\cdot 26^{-1} + ... + a_{-p}\cdot 26^{-p}$$

דוגמאות:

- ערכו של הביטוי "a.a" הינו 0.
- ערכו של הביטוי "ba.c" הינו $1\cdot 26^1 + 2\cdot 26^{-1} = 26.077$

השלימו את הפונקציה הבאה, המקבלת כקלט מחרוזת s המייצגת מספר חיובי בבסיס 26, ומחזירה את ערכו של המספר כ-double. ניתן להניח כי s מכילה קלט חוקי. **יש לבצע את החישוב במעבר יחיד על המחרוזת.**

```
double strval(char *s) {

  double base = 1, result = 0;

  while (*s != '.') {
    result *= 26;
    result += (*s - 'a');
    ++s;
  }

  while (*(++s) != 0) {
    base /= 26;
    result += (*s - 'a') * base;
  }

  return result;
}
```

סעיף ב (10 נקודות)

נתונים שני מערכים **a** ו-**b**. כל אחד מהם הינו מערך של מחרוזות, כאשר כל מחרוזת מייצגת מספר בבסיס 26 כמתואר בסעיף א'. שני המערכים **ממוינים** בסדר עולה, לפי הערך המספרי של המחרוזות שהם מכילים.

כתבו פונקציה המקבלת שני מערכים ממוינים **a** ו **b** כנ"ל, בגודל **n** ,**m** בהתאמה, ומספר ממשי **x**. הפונקציה מחזירה 1 אם קיימים זוג איברים **a[i]** ו-**b[j]** כך ש- **a[i]-b[j]==x** (שימו לב שהכוונה רק לאיבר מהמערך **a** פחות איבר מהמערך **b**, ולא להיפך). במידה ולא קיימים שני איברים כאלו הפונקציה תחזיר 0.

בשאלה זו נניח כי כל המחרוזות במערכים **a** ו-**b** הינן באורך לכל היותר c, כאשר c הוא קבוע ידוע. תחת הנחה זו, על הפונקציה שאתם כותבים בסעיף זה לעבוד בסיבוכיות זמן O(n+m), וסיבוכיות מקום נוסף O(1).

```
int finddiff(char* a[], int n, char* b[], int m, double x) {

  int i = 0, j = 0;
  double diff;

  while (i<n && j<m) {
    diff = strval(a[i]) - strval(b[j]);
    if (diff == x) return 1;
    if (diff < x) i++;
    else j++;
  }

  return 0;
}
```

שאלה 2 (20 נקודות)

נתונה מחרוזת המורכבת מאותיות אנגליות קטנות בלבד. **פרמוטציה** של מחרוזת זו מוגדרת כמחרוזת המכילה את אותו תוכן בדיוק, אולם ייתכן שבשינוי סדר (שימו לב שכל מחרוזת הינה פרמוטציה של עצמה).

לדוגמה,

- עבור המחרוזת "abc", פרמוטציות אפשריות הן "acb" , "cab" , "abc" (ויש פרמוטציות נוספות).
- עבור המחרוזת "abba", פרמוטציות אפשריות הן "baba", "aabb" , "abab" וכן הלאה.

השלימו את הפונקציה הבאה, המקבלת שתי מחרוזות **s1** ו-**s2**, ומחליטה האם הן פרמוטציה אחת של השנייה. הפונקציה מחזירה 1 אם כן, ו-0 אם לא. ניתן להניח ששתי המחרוזות חוקיות, ומכילות אותיות אנגליות קטנות בלבד (a-z).

על הפונקציה לעבוד בסיבוכיות זמן O(n+m) (כאשר n ו-m הינם אורכי שתי המחרוזות בהתאמה), וסיבוכיות מקום נוסף O(1).

<u>הערה</u>: ניתן להתייחס למספר האותיות בא"ב האנגלי כאל קבוע.

```
int permute(char* s1, char* s2) {

  int count[26] = {0}, i;

  while (*s1) {
    count[*s1 - 'a']++;
    s1++;
  }
  while (*s2) {
    count[*s2 - 'a']--;
    s2++;
  }

  for (i=0 ; i<26 ; ++i)
    if (count[i] != 0) return 0;
  return 1;
}
```

שאלה 3 (20 נקודות)

סעיף א (8 נקודות)

במהלך מבחן ב"מבוא למדעי המחשב", מתברר שהדופק של סטודנט ממוצע הינו:

- בדקה מספר 1 הוא 57 פעימות לדקה.
- בדקה מספר 2 הוא 60 פעימות לדקה.
- בכל דקה שהיא כפולה של 3, הדודה רושמת את השעה על הלוח, ולכן הדופק עולה ב-2 פעימות לעומת הדקה הקודמת. לדוגמא בדקה מספר 3 יהיה הקצב 60 + 2 = 62 פעימות לדקה.
- בכל דקה אחרת – הדופק הינו הממוצע של שתי הדקות הקודמות + 2. לדוגמא, בדקה מס' 4 הדופק יהיה הממוצע של הדקות 2 ו-3 ועוד 2, דהיינו 2 + [(62 + 60)/2] = 63.

השלימו את הפונקציה **`heartbeat()`** להלן, אשר מקבלת כפרמטר מספר דקה **`n`**, ומחזירה את הדופק של סטודנט ממוצע במבחן "מבוא למדעי המחשב" בדקה זו. על הפתרון להיות **רקורסיבי**.

כתבו את סיבוכיות **המקום** של הפונקציה שכתבתם כתלות ב-n (חסם Θ): **Θ(n)**

```
double heartbeat(int n) {

  if (n <= 1) return 57;
  if (n == 2) return 60;

  if (n%3 == 0)
    return heartbeat(n-1) + 2;
  else
    return (heartbeat(n-1)+heartbeat(n-2))/2 + 2;
}
```

סעיף ב (12 נקודות)

ידוע שסטודנט ממוצע מתעלף כאשר הדופק שלו מגיע (או עובר) את הערך m.
ממשו את הפונקציה **maxlen()** להלן, אשר מקבלת כפרמטר את הערך m (הדופק שהחל ממנו יתעלף הסטודנט), ומחזירה את הזמן המרבי של מבחן שניתן לתת מבלי שהסטודנט יאבד הכרתו.

סיבוכיות זמן: יש לפתור את השאלה תוך שימוש במספר מינימאלי של קריאות לפונקציה **heartbeat()** (דהיינו בסיבוכיות זמן אופטימאלית). כמובן שאין "להתחכם" על ידי הגדרת הפונקציה מחדש תחת שם אחר וכדומה.

רמז: שימו לב שהדופק הוא סדרה עולה-ממש, כלומר **heartbeat(n) ≥ heartbeat(n−1)+1**. כמו כן, שימו לב שהדופק בדקה ה-m גדול ממש מ-m: **heartbeat(m) > m**, ולכן מספר הדקה שאנו מחפשים הוא בהכרח קטן מ- m.

- ניתן להשתמש בפונקציה מסעיף א' גם אם לא פתרתם אותו.
- שימו לב שאם בדקה מסוימת n הדופק של הסטודנט **בדיוק** m, אז המבחן יכול להימשך רק n−1 דקות.

```
int maxlen(int m) {

 int min = 1, max = m, mid;
 double hb;

 if (m <= 57) return 0;

 while (max > min+1) {
   mid = (min + max)/2;
   hb = heartbeat(mid);
   if (hb == m) return mid - 1;
   if (hb < m) min_n = mid;
   if (hb > m) max_n = mid;
 }
 return min;
}
```

שאלה 4 (20 נקודות)

נתונה הפונקציה הבאה:

```
int func(long arr[], int n, long x)
{
  int half = n/2, i, j;
  int num1, num2;

  if (n==0) return 0;
  if (n==1) return (arr[0] <= x) ? 1 : 0;

  num1 = func(arr, half, x);
  num2 = func(arr+half, n-half, x);

  for (i=num1, j=half+num2-1 ;
       i<half && j>=half ; ++i, --j)
  {
    long tmp = arr[i];
    arr[i] = arr[j];
    arr[j] = tmp;
  }
  return num1 + num2;
}
```

א. רשמו את תוכן המשתנה **num** ואת תוכן המערך **arr** לאחר הרצת קטע הקוד הבא:

```
long arr[5] = {1, 8, 4, 7, 2};
int num = func(arr, 5, 4);
```

num :

3

arr :

1	2	4	8	7

ב. הסבירו בקצרה את הפעולה שמבצעת הפונקציה הנ"ל: בהנתן מערך **arr** וערך **x**, מה יכיל **arr** בתום הקריאה לפונקציה? ומה משמעות ערך ה-**int** אותו היא מחזירה?

הפונקציה func מסדרת את אברי המערך arr כך שכל האברים שקטנים או שווים ל-x יהיו בצד שמאל, וכל האיברים שגדולים מ-x יהיו בצד ימין (פעולה זו נקראית pivoting). הפונקציה מחזירה את מספר האברים שקטנים או שווים ל-x.

ג. כתבו את סיבוכיות הזמן והמקום של הפונקציה **func()** כתלות ב-**n**:

סיבוכיות זמן: Θ(**n log(n)**) סיבוכיות מקום נוסף: Θ(**log(n)**)

שאלה 5 (20 נקודות)

נתונים שני מערכים של **char** (שימו לב שאלו אינם מחרוזות):

- מערך בשם **arr[]** שאורכו **n**.
- מערך בשם **pat[]** (קיצור של pattern) שאורכו **m**.

השלימו את הפונקציה **printPatterns** (בעמוד הבא) שמקבלת את שני המערכים ואת אורכם, ומדפיסה למסך את כל המופעים של המערך **pat** בתוך המערך **arr**. **מופע של המערך pat בתוך המערך arr איננו חייב להיות רציף.**

לדוגמא, עבור הנתונים הבאים: **n = 10, m = 3**

המערך **arr** :

n	b	e	n	b	a	r	e	a	b

המערך **pat** :

n	b	a

המופעים של **pat** בתוך המערך **arr** הינם:

n	b	–	–	–	a	–	–	–	–
n	b	–	–	–	–	–	–	a	–
n	–	–	–	b	a	–	–	–	–
n	–	–	–	b	–	–	–	a	–
–	–	–	n	b	a	–	–	–	–
–	–	–	n	b	–	–	–	a	–

עבור הקלט הנ"ל, פלט הפונקציה שלכם צריך להראות כך:

```
nb---a----
nb------a-
n---ba----
n---b---a-
---nba----
---nb---a-
```

על הפתרון להיות רקורסיבי, ולעבוד בשיטת ה-backtracking.

לנוחותכם, ניתן להשתמש בפונקצית העזר הבאה, שמקבלת מערך של **char** בגודל **n** ומדפיסה את תוכנו (אין צורך לממש):

```
void printarray(char arr[], int n);
```

הערות נוספות: שימו לב שהפרמטר **k** המועבר לפונקציה **printPatternsAux** הוא לשימושכם לפי הצורך, ויש יותר מאפשרות אחת נכונה להשתמש בו. כמו כן, שימו לב כי מספר השורות הריקות בקוד לא בהכרח מעיד על מספר שורות הקוד שיש לכתוב, ואורך הקו אינו בהכרח מעיד על אורך השורה שיש להשלים.

```
void printPatterns(char arr[], int n, char pat[], int m)
{
  char *aux;  int i;

  aux = (char*)malloc(n);
  for (i=0; i<n; ++i) {
    aux[i] = '-';
  }

  printPatternsAux(arr, n ,pat, m, aux,     0     );
  free(aux);
}

void printPatternsAux(char arr[], int n, char pat[], int m,
                      char aux[], int k)
{
  if (m==0) {
    printarray(aux, n+k);
    return;

  }
  if (n==0) {
    return;

  }
  if (arr[0] == pat[0]) {
    aux[k] = arr[0];
    printPatternsAux(arr+1, n-1, pat+1, m-1, aux, k+1);
    aux[k] = '-';
  }
  printPatternsAux(arr+1, n-1, pat, m, aux, k+1);

}
```