מבוא למדעי המחשב מ' 234114

מבחן מועד ב', סמסטר אביב תשס"ד 27.9.2004

שם משפחה	שם פרטי	מס' סטודנט

משך המבחן: 3 שעות.
חומר עזר: אין להשתמש בכל חומר עזר.

שאלה	ערך	הישג
1	12	
2	15	
3	20	
4	15	
5	18	
6	20	
סכום	100	

הוראות לנבחנים ולנבחנות:
1 מלאו את הפרטים בראש דף זה (בעט).
2 בדקו שיש 13 עמודים (6 שאלות) כולל עמוד זה.
3 תשובותיכם ייכתבו על טופס המבחן בלבד.
4 כתבו בכתב-יד נקי וברור (מומלץ להשתמש בעפרון ומחק).
5 **אין להשתמש בפונקציות ספריה או בפונקציות שנלמדו בכיתה אלא אם צוין אחרת בשאלה.**
6 **ניתן להוסיף פונקציות עזר (כולל מימוש) כרצונכם.**
6 **אין לכתוב הסברים מפורטים לתשובות (אם לא התבקשתם לכך).**
7 **בכל השאלות ניתן להניח שהקלט תקין.**
8 ניתן לצבור עד 100 נקודות. נסו לצבור את מירב הנקודות.

אביב תשס"ד
מרצים: ד"ר רועי פרידמן, צחי קרני.
מתרגלים: אולג רוכלנקו, רון רובינשטיין, עזרא אוחיון, ניר זפקוביץ, ויסאם קאדרי.

שאלה 1 (12 נקודות)

1. נתונה פונקציה

```
int functionA(int *a, int n)
{
   int half;

   if (n <= 1) return 1;
   half = n/2;

   if ( functionA(a, half) && (a[half-1] <= a[half]) &&
          functionA(a+half, n-half) )
     return 1;
   else
     return 0;
}
```

מה מבצעת הפונקציה הנ"ל?_____

_____

_____

מהי סיבוכיות הזמן של הפונקציה?_____

מהי סיבוכיות המקום הנוסף של הפונקציה?_____

2. נתונה פונקציה

```
void functionB(int *a, int n)
{
   int *temp;
   int half, i;

   if (n<=1) return;

   half = n/2;
   functionB(a, half);
   functionB(a+half, n-half);

   temp = (int *)malloc(n*sizeof(int));

   for(i=0; i<n-half; i++)
      temp[i] = a[half+i];
   for(i=0; i<half; i++)
      temp[i+n-half] = a[i];
   for(i=0; i<n; i++)
      a[i] = temp[i];

   free(temp);
}
```

מה מבצעת הפונקציה הנ"ל?_____

_____

_____

מהי סיבוכיות הזמן של הפונקציה?_____

מהי סיבוכיות המקום <u>הנוסף</u> של הפונקציה?_____

שאלה 2 (15 נקודות)

א. בסעיף זה נרצה לחשב באופן יעיל את ערכו של פולינום נתון בנקודה נתונה. הפולינום יינתן לנו ע"י מערך מקדמים a (מסוג double), כאשר המקדם של האיבר ה-x^i נמצא בתא a[i].

לדוגמא: double a[4]={2,3,0,5}
מייצג את הפולינום $5x^3+3x+2$.

השלמו את הממוש של הפונקציה הבאה:

double calc_poly(**double*** a, **int** size, **double** x)

כאשר a הוא מצביע למערך המקדמים, size הוא גודל המערך ו-x הוא הנקודה אותה אנו מעוניינים לחשב. ניתן להניח ש-size>0.

דוגמאות: (עבור a שלמעלה) calc_poly(a, 4,1)=10, calc_poly(a, 4, 2)=5*8+3*2+2=48.

דרישה: על המימוש להיות רקורסיבי. אין להשתמש בפונקציות עזר, לולאות או משתנים סטטיים/גלובליים.

דרישות סיבוכיות: סיבוכיות זמן: O(size), סיבוכיות מקום O(size).

```
double calc_poly(double* a, int size, double x)
{
  if (size == 1 )
    return_______________

_____________________________________________________

_____________________________________________________

_____________________________________________________

}
```

ב. בסעיף זה נבצע העלאה יעילה בחזקה טבעית. השלמו את הממוש של הפונקציה הבאה:

unsigned long natural_power(**unsigned long** a, **unsigned long** b)

המקבלת שני מספרים טבעיים a ו-b, ומחזירה כתשובה את a^b.

דוגמאות: natural_power(2,10)=1024, natural_power(3,4)=81.

דרישות: על המימוש להיות רקורסיבי. אין להשתמש בפונקציות עזר, לולאות או משתנים סטטיים/גלובליים.

דרישות סיבוכיות: סיבוכיות זמן $O(\log b)$, סיבוכיות מקום $O(\log b)$.

```
unsigned long natural_power(unsigned long a, unsigned long b)
{
_____________________________________________________________
  if (b == 0)
    return_________________
  if (b == 1)
    return_________________

  if  ( b%2 == 0 )
_____________________________________________________________
_____________________________________________________________
  else
_____________________________________________________________
_____________________________________________________________
}
```

שאלה 3 (20 נקודות)

אחת הפעולות הנפוצות במחרוזות הינה הזזה מחזורית של אברי המחרוזת. לדוגמה: עבור המחרוזת "abc123def456", הזזה מחזורית שמאלה של המחרוזת בשלושה איברים תניב את המחרוזת "123def456abc". בסעיפים הבאים הינכם נדרשים לממש פעולת הזזה מחזורית שמאלה ב- i איברים במחרוזת בעלת n איברים (ניתן להניח כי $i \leq n$), ללא שימוש בפונקציות עזר, תחת המגבלות הנתונות.
בכל הסעיפים מותר להשתמש (אך לא חובה) בפונקציות swap, strlen, malloc.
<u>הערה:</u> שים לב כי i אינו קבוע לצורכי חישוב הסיבוכיות.

א. כאשר סיבוכיות הזמן הינה $O(n)$ וסיבוכיות המקום $O(n)$

void Cyclic_Shift_1 **(char*** str**, unsigned int** i**)**

ב. כאשר סיבוכיות הזמן הינה $O(n^2)$ וסיבוכיות המקום $O(1)$.

void Cyclic_Shift_2 **(char*** str**, unsigned int** i**)**

ג. בסעיף זה נשתמש בפעולה *reverse* אשר הופכת את סדר האיברים במחרוזת (לדוגמא עבור המחרוזת "abc123def456" תגרום הפעולה *reverse* לקבלת מחרוזת "654fed321cba") על מנת לבצע את פעולת ההזזה המחזורית שמאלה. ממשו את הפונקציה ()Reverse עם סיבוכיות זמן $O(n)$ וסיבוכיות מקום נוסף $O(1)$. והשתמשו בה על-מנת לבצע את פעולת ההזזה, בסיבוכיות זמן $O(n)$ וסיבוכיות מקום $O(1)$.

void Reverse **(char*** str, **const int** n**)**

רמז: שימו לב לזהות reverse(reverse(s) reverse(t)) = $t\ s$

כאשר s ו-t הן מחרוזות כלשהן ו-reverse(s) מסמן הפעלה של reverse על המחרוזת s.

void Cyclic_Shift_3 **(char*** str**, unsigned int** i**)**

שאלה 4 (15 נקודות)

בשאלה הזאת נדון בניסיון לשפר אלגוריתם quick-sort. מוצע להשתמש בשני איברי ציר (pivots) וע"י כך לשפר את סיבוכיות האלגוריתם.

פירוט השינוי: בכל רמה של רקורסיה נבחר שני איברי הציר. תחילה נחלק את איברי המערך לשתי קבוצות -- אלה שקטנים ואלה שגדולים מה-pivot הקטן מבין שני ה-pivot-ים. את האיברים מהקבוצה השניה שוב נחלק לשתי הקבוצות, הפעם ביחס ל-pivot הגדול.

א. להלן חלקי התוכנית המבצעת את המיון המוצע לעיל. עליכם להשלים את המקומות החסרים (בכל שורה מופיעה פקודה אחת). אין להשתמש בפונקציות נוספות פרט לפונקצית swap אשר נלמדה בכיתה.

```
void Sort( int a[ ], int n)
{
  int n0 = RandomVal(n), n1 = OtherRandomVal(n);
  int  b = 0,   t1 = n - 1, t0 = t1 - 1;
  int p0, p1;          /* איברי הציר */

  if (n < 2) return;

  swap(&a[n0], &a[0]);
  swap(&a[n1], &a[n-1]);
  if (a[n-1] < a[0]) swap(&a[0], &a[n-1]);
  p0 = a[0];  p1 = a[n-1];

  while (b <= t0) {
    while (____________________)_________________________________
    while (____________________)_________________________________
    if (b <= t0 ) swap(_____________________________________);
  }
  swap(_____________________________________________);

  while (b <= t1) {
    while (____________________)_________________________________
    while (____________________)_________________________________
    if (_____________) swap(_____________________________________);
  }
```

```
  swap(____________________________________________);

  Sort(____________________________________________);
  Sort(____________________________________________);
  Sort(____________________________________________);

  return;
}
```

ב. האם האלגוריתם מסעיף א' מהיר יותר מאלגוריתם quick-sort אשר נלמד בכיתה? אם כן – הסבירו. אם לא – הסבירו ותנו דוגמא למערך הקלט ולבחירת איברי הציר כך שסיבוכיות זמן הריצה של האלגוריתם מסעיף א' תהיה ריבועית בגודל המערך (בדוגמא אין צורך לפרט את כל צעדי הריצה אך יש להסביר מה מאפיין את בחירת איברי הציר שלכם).

שאלה 5 (18 נקודות)

הגדרה: *חציון* (median) של סדרת המספרים $(a_1, a_2, ..., a_n)$ (כאשר *n* אי-זוגי) הינו איבר a_k בקבוצה כך שמספר האיברים בקבוצה שקטנים או שווים ל- a_k שווה למספר האיברים בקבוצה שגדולים או שווים ל- a_k. עבור קבוצה עם מספר זוגי של איברים נגדיר את החציון בתור הקטן מבין שני המספרים האמצעיים.

דוגמאות:

- עבור הסדרה $(-10,3,3,7,-2)$ החציון הוא 3.
- עבור הסדרה $(1,7,6,2,5,9)$ החציון הוא 5.

נדרש: פתור את הסעיפים הבאים ללא שימוש בפונקציות ספריה או בפונקציות שנלמדו בכיתה.

א. נתון מערך מספרים שלמים A[n] הממויין בסדר עולה. כתוב פונקציה שתחזיר את החציון של קבוצת איברי המערך.

```
int Median (int A[], unsigned int n)
{
    ____________________________________________
    ____________________________________________
    ____________________________________________

}
```

ב. נתונים שני מערכים שלמים A[n] ו-B[n] , כל אחד מהם ממויין בסדר עולה (אך אין קשר בין איברים בשני המערכים).

מטרת הסעיף: לכתוב פונקציה שתחזיר את החציון של קבוצת האיברים של שני המערכים. הפונקציה תעבוד בזמן $O(\log n)$. בסעיף זה מותר (אך לא חובה) להשתמש ברקורסיה.

לדוגמא: עבור מערכים $A = \{3,5,6,7,8\}$ ו- $B = \{1,2,3,4,6\}$ הפונקציה תחזיר 4 כי זהו האיבר הקטן מבין שני האיברים האמצעיים בקבוצה בת 10 מספרים $\{1,2,3,3,4,5,6,6,7,8\}$.

תארו בקצרה את האלגוריתם (שימו לב שאם האלגוריתם המתואר אינו נכון, הקוד לא ייבדק):

_____

_____

_____

_____

_____

מימוש:

```
int Median2 (int A[],int B[], unsigned int n)
{
```

שאלה 6 (20 נקודות)

בחנות ישנם N גלילים של חבלים באורכים שונים (כל החבלים הם מאותו סוג וטיב). לכל גליל אורך משלו. אורכי הגלילים נתונים ע"י המערך Ropes[N]. לחנות הגלילים יש M הזמנות לאורכים שונים של חבלים. ההזמנות נתונות ע"י המערך Orders[M]. עבור כל אחת מ-M ההזמנות צריך להחליט מאיזה גליל היא תסופק.

לדוגמה: Ropes=(80,65,20) , Orders=(50,65,17,16).

את ההזמנה הראשונה (50) ניתן לספק מהגליל הראשון, את ההזמנה השנייה (65) ניתן לספק מהגליל השני, את ההזמנה השלישית (17) ניתן לספק מהגליל הראשון ואת ההזמנה הרביעית (16) ניתן לספק מהגליל השלישי.

את הפתרון לבעיה נספק ע"י המערך Cut[M] אשר כל כניסה i בו מתאימה להזמנה i במערך Orders[M] והיא מציינת את מספר הגליל שממנה תסופק אותה הזמנה i.

- בדוגמה שלנו : Cut[M]={0,1,0,2}.

לא תמיד ניתן לספק את כל ההזמנות (למשל אם הייתה הזמנה לחבל באורך 90 לא ניתן היה לספק אותה כי אין לחנות גליל חבל באורך של 90 לפחות).

נדרש:

ממשו את הפונקציה (אשר פותרת את הבעיה שתוארה לעיל):

```
int CutRopes(const int Ropes[N], const int Orders[M], int Cut[M], int M)
```

במידה וניתן לספק את כל ההזמנות - הפונקציה תכניס ערכים המתאימים להקצאה אפשרית של גלילי החבלים ל- Cut[M] (כך שכל צבר ההזמנות Orders[M] יסופק) וערך החזרה של הפונקציה יהיה אחד.

במידה ולא ניתן לספק את כל צבר ההזמנות - הפונקציה תחזיר אפס (ולא תהיה משמעות לערכים שב-Cut[M]) .

על הפתרון להיות רקורסיבי, יש להשתמש ב-backtracking אם ניתן. מותר להגדיר פונקציות עזר חדשות, אך אין להשתמש בפונקציות ספריה או בפונקציות שנלמדו בכיתה ללא הגדרה מפורשת שלהן. כמוכן מותר להגדיר משתנים פנימיים לוקליים חדשים לפי הצורך (אך לא גלובליים). יש להניח כי קבוע N מוגדר באמצעות ה-define.

```
int CutRopes(const int Ropes[N], const int Orders[M], int Cut[M] , int M)
{
```