Technion Embraces the Future

מבוא למדעי המחשב מ' 234114
מבוא למדעי המחשב ח' 234117

מבחן מועד ב', סמסטר חורף תשס"ד 15.03.04

שם משפחה	שם פרטי	מס' סטודנט

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

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

הוראות לנבחנים ולנבחנות:
1 מלאו את הפרטים בראש דף זה (בעט).
2 בדקו שיש 14 עמודים (6 שאלות) כולל עמוד זה ונספח.
3 התשובות ייכתבו על טופס המבחן.
4 כתבו בכתב-יד נקי וברור (מומלץ להשתמש בעפרון ומחק).
5 אין לכתוב הערות והסברים לתשובות.
6 בכל השאלות ניתן להניח שהקלט תקין.

חורף 234114
מרצים: ד"ר יחיאל קמחי, צחי קרני
מתרגלים: שאדי סאבא, אורית עדן, ניר זפקוביץ', עמיר אדלר, רן רובנשטיין, רוסטם טייגר.

חורף 234117
מרצים: ד"ר יחיאל קמחי, יואב ציבין, רועי מלמד.
מתרגלים: שאדי סאבא, אבישי טל, בני גודלין, חיה זלברשטיין, עזרא אוחיון, עידו פלדמן, נלה גורביץ'

שאלה 1 (15 נקודות)
סעיף א.

```
#define N 10
void crazy(int numbers[])
{
int i, j;
int m, temp;
for (i = 0; i < N-1; i++) {
    m = i;
    for (j = i+1; j < N; j++){
      if (numbers[j] < numbers[m])
        m = j;
    }
    temp = numbers[i];
    numbers[i] = numbers[m];
    numbers[m] = temp;
    }
}
```

1. איך יראה המערך numbers הבא אחרי הרצת הפונקציה ()crazy עליו?

לפני:

0	**3**	**4**	**5**	**7**	**6**	**8**	**9**	**2**	**1**

אחרי:

1. מהי סיבוכיות האלגוריתם (זמן ומקום נוסף) כפונקציה של N? _____

סעיף ב

נתונים המערכים/משתנים הבאים בזיכרון המחשב:

```
char * a          =  "BeaTles";
char *c[ ]        =  {"ABCDE","EBCD","ABAB","CDAB","ABB"};
int b[ ]          =  {2,0,1,3};

struct complicated {
        int a;
        char * c;
        int *d[2];
};

typedef struct complicated Complicated;
Complicated com;
com.a=b[a[7]];
(&com)->c=c[1];
com.d[0]=b+1;
com.d[1]=b;
```

כתבו את ערך הביטויים הבאים :

com.c[0]	
(&com)->c[com.a];	
com.a	
*com.d[b[b[0]]]	

כתבו מה ידפיסו הפקודות הבאות :

printf("%s",c[(&com.d[0][0])[1]]);	
printf("%s",(&com)->c+2);	

שאלה 2 (15 נקודות)

בכתה יש N תלמידים (N מוגדר ע"י #define)
לכל תלמיד יש:
שם (שאורכו לכל היותר 32 אותיות)

תעודת זהות (אשר תיוצג ע"י מספר מטיפוס long, תעודות הזהות יהיו בעלות 5 ספרות)
ציון במבחן (מטיפוס unsigned int בטווח 1-100).

עליכם לכתוב פונקציה **printLex** שמקבלת מבנה של כיתה ומדפיסה את שמות התלמידים, מספר תעודת הזהות שלהם וציונם כאשר סדר ההדפסה הוא הסדר הלקסיקוגרפי (א"ב) של שמות התלמידים (כל תלמיד בשורה נפרדת), השתמשו בפונקציה strcmp:
int strcmp(const char*, const char*)

על הפונקציה לעבוד בסיבוכיות זמן אופטימלית, **הקפידו על תכנות מבני.**

סעיף א
השלימו את ההגדרה של המבנה Student

```
struct Student
{
_____________________________________________
_____________________________________________
_____________________________________________
_____________________________________________
};
```

סעיף ב
בהנחה שהגדרתם מבנה של תלמיד struct student, נגדיר מבנה של כיתה struct Class .

```
struct Class
{
     struct Student *stud_array[N];
};
```

ממשו את הפונקציה printLex

```
void printLex(struct Class* class);
```

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

שאלה 3 (15 נקודות)

נתונה ההגדרה הרקורסיבית של סידרת טרי-בונאצי:

$$\boxed{\begin{aligned} a_0 &= 1 \\ a_1 &= 2 \\ a_2 &= 3 \\ a_n &= a_{n-3} + a_{n-2} + a_{n-1} \end{aligned}}$$

למשל 10 האיברים הראשונים הם:

230 125 68 37 20 11 6 3 2 1

סעיף א

השלם את הפונקציה הרקורסיבית part1_TriFib שמחזירה את a_n:

```
unsigned int part1_TriFib(unsigned int n) {
  if (_______) return ________;
  return        ___________________________________
        ___________________________________________
        ___________________________________________;
}
```

מהי סיבוכיות הזמן והמקום של part1_TriFib כפונקציה של n?

זמן: O(_____) מקום: O(_____)

סעיף ב

השלם את ההגדרה הרקוסיבית החדשה (והיותר יעילה) של part2_TriFib:

```
unsigned int part2_TriFib_aux(unsigned int n, int a, int b, int c) {
  if (n==0) return a;
  return        ___________________________________
        ___________________________________________
        ___________________________________________;

}
unsigned int part2_TriFib(unsigned int n) {
  return ___________________________________________;
}
```

מהי סיבוכיות הזמן והמקום של part2_TriFib כפונקציה של n?

זמן: O(_____) מקום: O(_____)

סעיף ג

השלם את ההגדרה האיטרטיבית (כלומר ללא שימוש ברקורסיה) של part3_TriFib:

```
unsigned int part3_TriFib(unsigned int n) {
  int a=1, b=2, c=3, tmp;
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  return a;
}
```

מהי סיבוכיות הזמן והמקום של part3_TriFib כפונקציה של n?

זמן: O(_____) מקום: O(_____)

שאלה 4 (15 נקודות)

ועדת מיוחדת שהוקמה על ידי ממשלת ישראל החליטה, לאחר חמש שנות דיונים, כי יש להפסיק להשתמש באלגוריתמי מיון הקיימים ויש להשתמש מעכשיו באלגוריתם מיון חדש שיכונה מיון אינדקסים.
להלן קטע מדיוני הועדה:
.... כאשר נמיין מערך בשימוש במיון אינדקסים נשאיר את המערך המקורי ללא שינוי ונייצר מערך נוסף (הממיין) אשר במקום ה-i יכיל את האינדקס של האיבר ה-i בגודלו מהמערך המקורי.

דוגמא:
נתון המערך הבא : a = 9 8 -2 2 0
לאחר מיון אינדקסים נקבל את המערך הממיין הבא: 2 4 3 1 0
ואכן במערך a נמצא את האיבר הקטן ביותר (-2) במיקום a[2] ואת האיבר הבא (0) נמצא במקום a[4] ....

סעיף א
השלם את הפונקציה הבא הממשת מיון אינדקסים: (יש לדאוג לסיבוכיות מקום נוסף O(1), סיבוכיות זמן O(n^2))

```
void index_sort(int a[], int n, int sorted_indexes[])
{
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
_____________________________________________________
}
```

סיבוכיות זמן: _____

סעיף ב

השלם את הפונקציה הבאה המאתרת ביעילות איבר במערך לא ממוין (ע"י שימוש בסעיף א)

```
int find(int a[], int n, int x)
{
     int low, mid, high;
     int* sorted_indexes = ______________________________
     if(sorted_indexes ==  _________________   )
     {
          ___________________________
     }
     index_sort(a, n, sorted_indexes);
     low =  _____________;
     high = _____________;

     while( ________________)
     {









          return mid;
     }
     ___________________________________
       return -1;
}
```

סיבוכיות זמן: _____

סיבוכיות מקום נוסף: _____

<u>שאלה 5 (20 נקודות)</u>

נתון מערך A בגודל 2<n שמקיים A[i-1]≠A[i] לכל 1≤i≤n-1.

הגדרות:
אינדקס i, 0<i<n-1, יקרא:

- נקודת *מקסימום* אם A[i-1]<A[i]>A[i+1]
- נקודת *מינימום* אם A[i-1]>A[i]<A[i+1]
- נקודת *אקסטרימום* אם האינדקס הוא נקודת *מינימום* או *מקסימום*
- נקודת *עלייה* אם A[i-1]<A[i]<A[i+1]
- נקודת *ירידה* אם A[i-1]>A[i]>A[i+1]

אינדקס i=0 יקרא נקודת *עלייה* אם A[0]<A[1] ואחרת הוא יקרא נקודת *ירידה*.
אינדקס i=n-1 יקרא נקודת *עלייה* אם A[n-2]<A[n-1] ואחרת הוא יקרא נקודת *ירידה*.

בכל הפונקציות בכל הסעיפים בשאלה זו תמיד מעבירים בנוסף למערך A גם את גודלו n.

<u>סעיף א</u>
השלם את הפונקציה classify כך שבהנתן אינדקס i במערך A תחזיר האם האינדקס הוא נקודת *עלייה* (UP), *ירידה* (DOWN), או *אקסטרימום* (EXTREME).

typedef enum {UP, EXTREME, DOWN} CLASSIFICATION;

```
CLASSIFICATION classify(int A[], int n, int i) {
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
        ____________________________________________
}
```

<u>סעיף ב</u>

נניח ש-

1. A[0] = A[n-1] = 0

שימו לב שעכשיו מובטחת לנו <u>קיומה</u> של נקודת *אקסטרימום* מכיוון שהנחנו בתחילת השאלה ש- 2<n וגם ש- A[i-1]≠A[i].

2. שכל איברי המערך הם כולם חיוביים או כולם שליליים.

שימו לב שתחת הנחה זו מתקיים שאינדקס 0 הוא נקודת *עלייה* ואינדקס n-1 הוא נקודת *ירידה*, <u>או להיפך</u> (כלומר אינדקס 0 הוא נקודת *ירידה* ואינדקס n-1 הוא נקודת *עלייה*).

השלם את הפונקציה find_extrim שמקבלת מערך A ומחזירה נקודת *אקסטרימום* כלשהיא.

כל הפונקציות צריכות לרוץ בסיבוכיות זמן O(log n) וסיבוכיות מקום O(1).
ניתן להשתמש בפונקציות שהוגדרו בסעיפים הקודמים.

```
int find_extrim(int A[], int n) {
     int low=0, mid, high=n-1;
     CLASSIFICATION class_low, class_mid, class_high;
     ____________________________________________________
     ____________________________________________________
     ____________________________________________________
     ____________________________________________________

     while (low < high) {
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
          ____________________________________________
     }
     return -1;
     }
```

שאלה 6 (20 נקודות)

נתון הטיפוס **מישור** כמערך דו-מימדי `typedef int plane[ROWS][COLS]`

(בתכנית מוגדרים `ROWS,COLS` ע"י `#define`). מתיחסים לערכים במערך כאל נקודות גובה

(plane[i][j]>0 לכל i,j בתחום).

נקודת שיא: היא איבר במטריצה ש**אין** לו שכן (8 לכל היותר) גדול ממנו

(מותר להניח שאין שכנים שווים בגודלם, אך זה אינו משפיע על קושי הבעיה).

מסלול עולה: הוא סדרה של איברים במערך, $\square_{\llbracket}$ ϕ $\lightning$ $\phi_{\circledast}\square$ $\phi_{\otimes}\square$

כך שכל $\square_{\wr\ominus\circledast}$ הוא שכן של $\square_{\wr}$ וגם **גדול ממש** ממנו.

נקודת שיא קרובה ביותר: המסלול העולה אליה הוא הקצר ביותר מנקודת מוצא נתונה.

סעיף א

כתבו פונקציה רקורסיבית (רגילה) המקבלת מישור ונקודה בו (קואורדינטות אורך ורוחב) ומחזירה **ערך** של נקודת-שיא כלשהי ואת **מקומה** של הנקודה הזו (קואורדינטות אורך ורוחב).

השימוש בלולאות אסור.

<u>סעיף ב</u>

כתבו פונקציה, עם אותה הצהרה כמו הפונקציה מסעיף א', המשתמשת בטכניקה של **ניסוי וטעייה** (**Back-Tracking**) כדי למצוא את נקודת השיא **הקרובה ביותר**.

(בונוס משמעותי לכל הכותב פתרון נכון, שבו האלגוריתם יעיל ממש מהפתרון הנכון הפשוט)