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

סמסטר חורף תשע"ד

מבחן מסכם מועד א', 20 פברואר 2014

מספר סטודנט	*שם משפחה*	*שם פרטי*

משך המבחן: 3 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 19 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 לכתיבת תשובותיכם. סמנו טיוטות באופן ברור על מנת שהן לא תיבדקנה.
- יש לכתוב באופן ברור, נקי ומסודר. ניתן בהחלט להשתמש בעיפרון ומחק, פרט לדף השער אותו יש למלא בעט.
- בכל השאלות, הנכם רשאים להגדיר ולממש פונקציות עזר כרצונכם. לנוחיותכם, אין חשיבות לסדר מימוש הפונקציות בשאלה, ובפרט ניתן לממש פונקציה לאחר השימוש בה.
- בשאלה 4 בלבד, ניתן להשאיר פתרון ריק ולקבל עבורו 20% מהניקוד.

הנחיות תכנות כלליות, אלא אם מצוין אחרת בשאלה:

- אין להשתמש בפונקציות ספריה 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: גב' אנסטסיה דוברובינה, ד"ר רן רובינשטיין, פרופ' רון קימל (מרצה אחראי).

מתרגלים: נמרוד סבן פרטוש, אריק יודין, נדיה ליבאי, אביב סגל, בת-חן גולדן, רן ברנשטיין, דור הריס, ברק פת, נחשון כהן (מתרגל אחראי).

בהצלחה!

שאלה 1 (25 נקודות)

בכל אחד מקטעי הקוד הבאים, חשבו את סיבוכיות הזמן והמקום של הפונקציה f כתלות ב-n.

לנוחיותכם, להלן תזכורת לחוקי לוגריתמים:

$\log(a \cdot b) = \log(a) + \log(b)$

$\log(a/b) = \log(a) - \log(b)$

$\log(a^b) = b \cdot \log(a)$

$\log_a(a^b) = b$

א.

```
int f(int n)
{
   int k=0;
   while (n>0)
   {
       k += n;
       n--;
   }

   int res=0;
   while (k>0)
   {
       k /= 4;
       res++;
   }

   return res;
}
```

סיבוכיות זמן: Θ (n) ___ סיבוכיות מקום: Θ (1)

ב.

בסעיף זה, הפונקציה `bin_search()` מבצעת חיפוש בינארי במערך באורך n. הפונקציה מחזירה את האינדקס בו נמצא האיבר במערך, או 1- אם אינו נמצא. הפונקציה בעלת סיבוכיות זיכרון O(1).

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


int f(int a[], int n)
{
   int i=1, x=1;
   while (i<n)
   {
      if (bin_search(a,i,x) >= 0)
      {
         return x;
      }
      i *= 2;
      x *= 2;
   }

   return 0;
}
```

סיבוכיות זמן: Θ ($\log^2(n)$) סיבוכיות מקום: Θ (1)

ג.

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

int g(int n, int k)
{
   if (n < 2)
   {
       return 1;
   }

   int x = aux(n) + aux(k);
   return x + g(n/2,k*2);
}

int f(int n)
{
   return g(n,1);
}
```

סיבוכיות זמן: Θ ($\log^2(n)$) סיבוכיות מקום: Θ ($\log(n)$)

שאלה 2 (25 נקודות)

כתבו פונקציה המקבלת מערך של מספרים שלמים a ואת אורכו n, ומוצאת את רצף האיברים במערך שסכומו מקסימלי. הפונקציה מחזירה את סכום הרצף שנמצא, וכן כותבת את האינדקס בו מתחיל הרצף ואת אורכו לזוג משתני פלט id ו-len (בהתאמה) שמועברים לפונקציה באמצעות מצביעים. לדוגמה, עבור המערך הבא:

-5	3	2	-1	5	-8	-2

הרצף המקסימלי הוא הרצף (3, 2, -1, 5) שסכומו 9. הרצף מתחיל באינדקס 1 ואורכו 4. לפיכך, הפונקציה תחזיר 9 כערך ההחזרה, וכן תכתוב 1 למשתנה הפלט id ו-4 למשתנה הפלט len.

על הפונקציה לעמוד בסיבוכיות זמן $O(n^2)$ וסיבוכיות מקום נוסף $O(1)$.

הערות:

- ייתכן יותר מרצף אחד שסכומו מקסימלי. במקרה זה יש להחזיר רצף כלשהו שסכומו מקסימלי.
- ייתכן כי הפתרון יהיה רצף באורך 0 (למשל, אם כל המספרים במערך שליליים). במקרה זה יש להחזיר את הערך 0 כסכום המקסימלי, וכן לכתוב 0 לשני משתני הפלט.

```
int maxseq(int a[], int n, int* id, int* len) {
	int max=0;
	*id=0;
	*len=0;
	//maximum subsets for all possible starting points
	for(int i=0; i<n; ++i){
		int cur len;
		int cur max = max fixed start(a+i, n-i, &cur len);
		if(cur max>max){
			max = cur max;
			*id = i;
			*len = cur len;
		}
	}
	return max;
}

//find max subset starting at the first element
//time: O(n)
int max fixed start(int a[], int n, int *len){
	int sum=0, max=0;
	for(int i=0; i<n; ++i){
		sum+=a[i];
		if(sum>max){
			max=sum;
			*len = i+1;
		}
	}
	return max;
}
```

```
//Another solution in O(n).
int maxseq(int a[], int n, int* id, int* len) {
   //sumMaxSum contains the maximum subset that contains the
    //current value a[i]
    int maxSum = 0,  subMaxSum = 0, subSumStart = 0;
    *len = 0, *id = 0;
    for (int i = 0; i < n; ++i) {
     if (subMaxSum < 0) {//fresh start.
          subMaxSum = a[i];
          subSumStart = i;
     } else {//continue with max subset containing a[i-1]
          subMaxSum += a[i];
     }

     if (subMaxSum > maxSum) {
          maxSum = subMaxSum;
          *id = subSumStart;
          *len = i - subSumStart + 1;
     }

    }

    return maxSum;

}
```

שאלה 3 (25 נקודות)

בממלכה רחוקה ישנן N ערים, ובכל עיר בדיוק K תושבים (כאשר N ו-K מוגדרים כ-#define). בממלכה קיימת מערכת גביית מיסים המאחסנת את הנתונים הבאים: המערך **a[N]** מכיל את סכומי המס שמשלמת כל עיר, כאשר **a[i]** הוא הסכום שמשלמת העיר ה-i. המערך **id[N][K]** מכיל את מספרי תעודת הזהות של תושבי כל עיר, כאשר השורה ה-i של המערך, **id[i][0..K-1]** מכילה את מספרי תעודת הזהות של התושבים בעיר ה-i.

בשאלה זו נרצה לכתוב פונקציה הממיינת את הערים על פי סכום המס שהן משלמות, בסדר עולה. עליכם לממש פונקציה (בעמוד הבא) המקבלת את המערכים **a** ו-**id**, וממיינת אותם כך שלאחר המיון המערך **a** יהיה בסדר עולה. בתהליך זה יש לשמור על ההתאמה בין המערכים, כלומר, גם לאחר המיון השורה ה-i במערך **id** צריכה להכיל את רשימת התושבים בעיר שמשלמת מס בגובה **a[i]**.

דרישות סיבוכיות: כיוון שבסבירות גבוהה K גדול מ-N, נרצה סיבוכיות קטנה ככל האפשר ביחס ל-K. לפיכך, דרישות הסיבוכיות הינן: **זמן** $O(N^2 + NK)$, **זיכרון נוסף** $O(1)$.

הערות:

- בשאלה זו הערכים N ו-K אינם נחשבים כקבועים לצרכי סיבוכיות.
- יש לכתוב במקום המתאים למטה את סוג המיון בו השתמשתם, ולהסביר בקצרה מדוע הוא עומד בדרישות הסיבוכיות.
- פתרון שאינו עומד בדרישות הסיבוכיות יקבל ניקוד חלקי בלבד.

אלגוריתם המיון שהשתמשתי בו:max_sort _____

הסבר לסיבוכיות:

הסיבוכיות של swap קבועה. הסיבוכיות של find_max_idx היא O(n) כי יש לולאה אחת מ 1 עד n.

הסיבוכיות של swap_rows היא O(k) כי יש לולאה אחת מ 1 עד k.

הפונקציה sort מפעילה N פעמים את הפונקיות xdfind_max_i עם פרמטר קטן מ N, ואת הפונקציה swap_rows עם פרמטר K, ולכן זמן ריצה של כל איטרציה חסום ב O(N+K), וזמן ריצה כולל O(N*(N+K). סיבוכיות מקום O(1) כי לא השתמשנו ברקורסיה או מערכים. _____

```
void sort(int a[N], int id[N][K]) {
   for(int i=N; i>1; i--){
      int max=find max idx(a, i);
      swap(a+max, a+i-1);
      swap rows(K, id[max], id[i-1]);
    }
}
void swap(int *a, int *b){
      int tmp=*a;
      a=b;
      *b=tmp;
}
void swap rows(int k, int town1[], int town2[]){
      for(int i=0;i<k; ++i)
            swap(town1+i, town2+i);
}
int find max idx(int a[], int n){
      int max = 0;
      for(int i=1; i<n; ++i)
            if(a[i]>a[max])
                  max=i;
      return max;
}
```

שאלה 4 (25 נקודות)

בשאלה זו נממש פונקציה שמקבלת סכום כסף נתון sum, ומחזירה את מספר הצורות השונות שבהן ניתן להגיע לסכום כסף זה באמצעות צירופים של מטבעות. ערכי המטבעות הזמינים לנו נמצאים במערך []coins שאורכו n, כאשר כל ערך של מטבע מופיע במערך פעם אחת בדיוק. ניתן להניח שהמערך []coins מכיל ערכים שלמים, חיוביים, ושונים זה מזה. שימו לב שניתן להשתמש בכל מטבע מספר לא מוגבל של פעמים על מנת להגיע לסכום sum.

לדוגמה, עבור המערך []coins הבא:

```
int coins[2] = {1, 2};
```

ועבור הסכום sum=4, הפונקציה תחזיר 3 כיוון שניתן להגיע לסכום 4 באמצעות צירופי המטבעות הבאים:

1 + 1 + 1 + 1 = 4
1 + 1 + 2 = 4
2 + 2 = 4

שימו לב: בשאלה זו אין משמעות לסדר המחוברים. כלומר, הסכומים (2+1+1), (1+2+1) ו-(1+1+2) נחשבים לאותו הפירוק, ויש לספור אותם פעם אחת בלבד.

יש לפתור את השאלה ברקורסיה. בשאלה זו אין דרישות סיבוכיות.

```
int change(int coins[], int n, int sum) {
        if(sum==0)
                return 1;
        if(n==0 || sum<0)
                return 0;
        return
                change(coins, n, sum-coins[0])+ //pick coin 0
                change(coins+1, n-1, sum);// don't pick coin 0
}
```