תאריך הבחינה: 21.12.2012

שמות המרצים: פרופ' משה זיפר
ד"ר פז כרמי
מר עדי סוויסה
פרופ' מייק קודיש
מר אילן קידר
ד"ר צחי רוזן

שם הקורס: מבוא למדעי המחשב

מספר הקורס: 202-1-1011

שנה: 2013 סמסטר: א' מועד: בוחן

משך הבחינה: **2.5 שעות**

חומר עזר: **אסור**

<u>בוחן אמצע</u>

אנא קיראו היטב את ההוראת שלהלן:

- במבחן זה 4 שאלות. ענו על כל השאלות. בשאלות בהן לא מצוין אחרת, ניתן לבחור בפתרון רקורסיבי או פתרון שאינו רקורסיבי, לבחירתכם.
- רשמו את תשובותיכם **בדפי התשובות בלבד**. המחברת שקיבלתם היא מחברת טיוטה והיא לא תימסר כלל לבדיקה. בסיום הבחינה נשמור אך ורק את דף התשובות. כל שאר החומר יועבר לגריסה.
- **שימו לב:** החשיבות העליונה היא **לנכונות** הקוד. מאידך, **<u>יעילות</u>, סגנון וכתיבה ברורה** חשובים גם הם, ולכן תשובה יעילה ומסוגננת תזכה בציון גבוה יותר.
- <u>בשאלות התכנות</u>, מספר השורות העומדות לרשותכם בדף התשובות רומז על אורך הקוד הנדרש. הקפידו על כתב יד ברור. **תשובות מסורבלות או ארוכות מדי לא יזכו בניקוד מלא**. כתבו פתרון לשאלה קודם במחברת הטיוטה ורק לאחר מכן העתיקו פתרון נקי לדף התשובות.
- אין להוסיף פונקציות עזר אלא אם נאמר במפורש שמותר.
- אם יש סעיף בשאלה המסתמך על סעיף אחר, מותר להשתמש בו גם אם לא פתרתם את הסעיף האחר.
- **הקפידו לרשום בשלושת דפי התשובות גם את מספר הנבחן ומספר החדר שבו אתם נבחנים.**
- **במידה ואינכם יודעים את התשובה לסעיף כלשהו, רשמו "לא יודע/ת" (במקום תשובה) ותזכו ב- 20% מניקוד הסעיף. אם רשום "לא יודע/ת", ההתייחסות היא לכל הסעיף.**

<u>שאלה 1 (20 נקודות)</u>

נאמר כי מערך של מערכים int[][] matrix הינו מטריצה אם: matrix != null, matrix.length>0, לכל אינדקס i בתחום המערך matrix[i].length==matrix.length. הערך n=matrix.length נקרא המימד של המטריצה.

בהינתן מטריצה int[][] matrix מימד n, ותא matrix[i][j] במטריצה, אנו נתעניין בתאים הסימטריים לו תחת סיבוב המטריצה ב-90 מעלות, 180 מעלות, ו-270 מעלות. נאמר שארבעת התאים הם סימטריים תחת סיבוב. למשל, במטריצה ממימד 8, התאים matrix[1][2], matrix[2][6], matrix[6][5], ו- matrix[5][1] הם סימטריים תחת סיבוב. (כמו בציור הבא).

		{1,2}					
						{2,6}	
	{5,1}						
					{6,5}		

כתבו פונקציה void sym(int[][] matrix, int i, int j) אשר בהינתן מטריצה matrix וכן אינדקסים i ו-j אשר מציינים תא חוקי במטריצה, מבצעת השמה של ארבעת התאים הסימטריים תחת סיבוב (כולל התא matrix[i][j]) לערכים 1,2,3,4 בסדר כלשהו. הפונקציה אינה משנה את הערכים בשאר התאים במטריצה.

ניתן להניח כי matrix הינה מטריצה ממימד זוגי על-פי ההגדרה בתחילת השאלה, וניתן להניח ש-i,j מייצגים אינדקסים חוקיים במטריצה.

לדוגמא: עבור המטריצה מימין (ממימד 6), והקלט i=4, j=2 לאחר הפעלת הפונקציה, המטריצה יכולה להיראות כמו המטריצה משמאל:

	0	1	2	3	4	5
0						
1				3		
2		2				
3					4	
4			1			
5						

	0	1	2	3	4	5
0						
1						
2						
3						
4						
5						

שאלה 2 (27 נקודות)

א. (12 נקודות) בניו יורק בניין בעל n קומות. ספיידרמן רוצה לטפס על הבניין ולהגיע מהרחוב אל הקומה האחרונה. ספיידרמן יכול לטפס בכל צעד קומה אחת או שתיים.
כתבו פונקציה `int spiderman(int n)` אשר מחשבת בכמה דרכים שונות יכול ספיידרמן להגיע לקומה האחרונה של הבניין.

לדוגמא, קריאה לפונקציה `spiderman(4)` תחזיר ערך 5 מכיוון שבבניין בעל 4 קומות יכול ספיידרמן להגיע לקומה האחרונה ב-5 דרכים שונות:

1. לטפס קומה אחת בכל פעם עד שיגיע לקומה הרביעית
2. לטפס לקומה ראשונה, אח"כ לקומה שנייה ובצעד השלישי לדלג שתי קומות לקומה הרביעית.
3. לטפס לקומה ראשונה, אח"כ לשלישית ובצעד השלישי לרביעית.
4. לטפס לקומה השנייה, אח"כ לשלישית ובצעד השלישי לרביעית.
5. בצעד הראשון לטפס לקומה השנייה ובצעד השני לטפס לרביעית.

ב. (15 נקודות) בבניין נתון יש n קומות ($n < 30$), ובקומה ה-20 יש תאי טלפון בהם יכול ספיידרמן להוריד את התחפושת ולהפוך לפיטר פרקר ומשם לקחת את המעלית לקומה האחרונה.
כתבו פונקציה בשם: `int spidermanPhoneBooth20(int n)` אשר
מחשבת בכמה דרכים שונות יכול ספיידרמן להגיע לקומה האחרונה (קומה n) בבניין שבו יש מעלית ישירה לקומה האחרונה בקומה 20 . שימו לב, כעת יש לספיידרמן קיצור דרך שהוא **חייב** להשתמש בו, כאשר הוא מגיע לקומה זו (הוא לא יכול להגיע לקומה 20 ולהמשיך ממנה בלי לקחת את המעלית). אתם רשאים להשתמש בקריאות לפונקציה מסעיף א' גם אם לא רשמתם אותה.

לדוגמא, שלוש דרכים שונות (מתוך X האפשרויות) לטפס על בניין עם 26 קומות (ומעלית בקומה ה-20):

1. לטפס קומה אחת בכל פעם עד שיגיע לקומה ה- 20 ומשם ישירות לקומה האחרונה.
2. לטפס קומה אחת בכל פעם עד שיגיע לקומה ה- 19, אח"כ לטפס לקומה 21 ומשם קומה אחת בכל פעם עד שיגיע לקומה האחרונה.
3. לטפס לקומה ראשונה, אח"כ לדלג שתי קומות בכל פעם (לקומה השלישית החמישית השביעית ...) עד שיגיע לקומה ה- 25 ומשם לטפס לקומה האחרונה.

דוגמה לטיפוס לא אפשרי:

1. לטפס קומה אחת בכל פעם עד שיגיע לקומה האחרונה.

טיפוס זה אינו אפשרי מכיוון שהוא לא יכול להגיע לקומה 20 ולהמשיך ממנה בלי לקחת את המעלית לקומה האחרונה.

<u>שאלה 3 (28 נקודות, 2 נקודות לסעיף)</u>

מיכאל, שם לב שהוא מרבה להשתמש בשורה: i = i + 1;.
למשל, הוא כתב את הקטע קוד הבא:

```
public static void main(String[] args) {
	int i = 5;
	i = i + 1;
	System.out.println("i="+i);
}
```

הוא החליט ליצור מספר פונקציות שיחלפו את השורה הנ"ל.
עבור כל סעיף רשמו האם התוכנית תתקמפל ואם כן, מה יודפס:

a.
```
public static int incI(int i) {i = i + 1; return i;}

public static void main(String[] args) {
	int i = 5;
	i = incI(i);
	System.out.println("i="+i);
}
```

b.
```
public static void incI(int i) {i = i + 1; return i;}

public static void main(String[] args) {
	int i = 5;
	i = incI(i);
	System.out.println("i="+i);
}
```

c.
```
public static int incI(int j) {j = j + 1; return j;}

public static void main(String[] args) {
	int i = 5;
	i = incI(i);
	System.out.println("i="+i);
}
```

d.
```
public static void incI(int[] a) {a[0] = a[0] + 1;}

public static void main(String[] args) {
	int i = 5;
	int[] a = new int[1];
	a[0] = i;
	incI(a);
	System.out.println("i="+i);
}
```

e.
```java
public static void incI(int[] a) {a[0] = a[0] + 1;}

public static void main(String[] args) {
	int i = 5;
	int[] a = new int[1];
	a[0] = i;
	incI(a);
	i = a[0];
	System.out.println("i="+i);
}
```

f.
```java
public static int incI(int[] a) {return new int[a.length + 1];}

public static void main(String[] args) {
	int i = 5;
	int[] a = new int[i];
	i = incI(a);
	System.out.println("i="+i);
}
```

g.
```java
public static int[] incI(int[] a) {return new int[a.length + 1];}

public static void main(String[] args) {
	int i = 5;
	int[] a = new int[i];
	a = incI(a);
	i = a.length;
	System.out.println("i="+i);
}
```

h.
```java
public static void incI(int i) {i = i + 1;}

public static void main(String[] args) {
	int i = 5;
	incI(i);
	System.out.println("i="+i);
}
```

i.
```java
public static void incI() {i = i + 1;}

public static void main(String[] args) {
	int i = 5;
	i = incI(i);
	System.out.println("i="+i);
}
```

j.
```java
public static int incI(int i) {i = i + 1;}

public static void main(String[] args) {
	int i = 5;
	incI(i);
	System.out.println("i="+i);
}
```

k.
```
public static int incI() {i = i + 1;}

public static void main(String[] args) {
       int i = 5;
       i = incI(i);
       System.out.println("i="+i);
}
```

l.
```
public static int incI(int i) {i = i + i; return i;}

public static void main(String[] args) {
       int i = 5;
       i = incI(i);
       System.out.println("i="+i);
}
```

m.
```
public static int incI(int i) {i = i + 1;}

public static void main(String[] args) {
       int i = 5;
       i = incI(i);
       System.out.println("i="+i);
}
```

n.
```
public static int incI(int i) {i = i + 1; return i;}

public static void main(String[] args) {
       int i = 5;
       incI(i);
       System.out.println("i="+i);
}
```

שאלה 4 (25 נקודות)

מעודכן

מיכאל החליט לפתח אלגוריתם מיון חדש. לצורך כך הוא החליט לשלב בין שני אלגורתמים שלמד בכיתה – מיון הכנסה (insertion sort) ומיון בחירה (selection sort). מיכאל כתב את הקוד הבא:

```
1.  public static void michaelSort(int[] arr) {
2.    for(int i=0; i<arr.length-1; i=i+1) {
3.         int minInd = minIndex(arr, i);
4.         insert(arr, minInd);
5.    }
6.  }
```

```
public static void insert(int[] arr, int i) {
      int value = arr[i];
      while (i > 0 && arr[i-1] > value) {
        arr[i] = arr[i-1];
        i = i-1;
      }
      arr[i] = value;
}

public static int minIndex(int[] arr, int from) {
      int ans = from;
      for(int i =from+1; i<arr.length; i=i+1)
            if (arr[ans] > arr[i])
                  ans = i;
      return ans;
}
```

א. (5 נקודות) האם בסוף הלולאה (בפונקציה: michaelSort(int[] arr)) המערך ממוין?

ב. (10 נקודות) מה סיבוכיות זמן הריצה (בסדר גודל) של האלגוריתם הנ"ל במקרה הגרוע.

ג. (10 נקודות, שתי נקודות לסעיף) ללא כל קשר לתשובתכם בסעיף א', עבור כל אחת מהטענות הבאות רשמו האם היא נכונה או לא (אין צורך להוכיח). הטענות מתייחסות לתחילת הלולאה בפונקציה: public static void michaelSort(int[] arr).

1. בכל שלב של הלולאה (שורה 2, לפני בדיקת תנאי הלולאה) האיברים [0..i-1] ממוינים.
2. בכל שלב של הלולאה (שורה 2, לפני בדיקת תנאי הלולאה) האיברים [i..n-1] קטנים (או שווים) מכל האיברים [0..i-1].
3. בכל שלב של הלולאה (שורה 2, לפני בדיקת תנאי הלולאה) האיברים [i..n-1] גדולים (או שווים) מכל האיברים [0..i-1].
4. בשורה 3 בכל שלב של הלולאה minIndex(arr, i) מחזיר אינדקס של איבר שגדול (או שווה) מכל האיברים [0..i-1].
5. בשורה 3 בכל שלב של הלולאה minIndex(arr, i) מחזיר אינדקס של איבר מינימאלי במערך.

נספח:

מיון בחירה (selection):

```
public static void selectionSort(int[] arr) {
	for(int i=0; i<arr.length-1; i=i+1) {
		int minInd = minIndex(arr, i);
		swap(arr, i, minInd);
	}
}

public static int minIndex(int[] arr, int from) {
	int ans = from;
	for(int i =from+1; i<arr.length; i=i+1)
		if (arr[ans] > arr[i])
			ans = i;
	return ans;
}

public static void swap(int[] arr, int i, int j)    {
	int tmp = arr[i];
	arr[i] = arr[j];
	arr[j] = tmp;
}
```

מיון הכנסה (insertion):

```
public static void insertionSort(int [] arr) {
	for (int i = 1; i<arr.length; i = i+1)
		insert(arr, i);
}

public static void insert(int[] arr, int i) {
	int value = arr[i];
	while (i > 0 && arr[i-1] > value) {
		arr[i] = arr[i-1];
		i = i-1;
	}
	arr[i] = value;
}
```