מבוא למדעי המחשב
202-1-101-1

סמסטר ב', תשס"ח, 2007/8

בוחן

ד"ר צחי רוזן
מר אילן כדר

משך הבחינה: שעתיים וחצי
חומר עזר: **אסור**

בבחינה 4 שאלות. **ענו** על כולן: הניקוד מסתכם ב-100 נקודות.

בשאלות התכנות החשיבות העליונה בד"כ היא על **נכונות** הקוד, מאידך, **יעילות וסגנון** חשובים גם הם, ולכן תשובה יעילה ומסוגננת תזכה בציון גבוה יותר מאשר תשובה מסורבלת או ארוכה מדי.

בכל השאלות אתם **רשאים** להשתמש בסעיפים קודמים, אם כי לא בהכרח תזדקקו לכך. אתם **רשאים** לעשות זאת אפילו אם לא עניתם על הסעיפים האמורים.

לבחינה מצורף בסופה **דף תשובות**. **רשמו** את תשובותיכם אך ורק בדף התשובות במקומות המיועדים. המחברת שקיבלתם היא מחברת טיוטה בלבד, והיא וטופס הבחינה עצמו **לא** יימסרו כלל לבדיקה.

שימו לב: מספר השורות הריקות בדף התשובות **בהחלט** מרמז על אורך התשובה.

הקפידו על כתב יד ברור.

הקפידו לרשום את מספר הנבחן על דף התשובות.

בהצלחה!

שאלה 1 (20 נקודות)

סעיף א (10 נקודות):

השלימו בדף התשובות את הפונקציה

```
boolean check (int [] vec, int c)
```

שמחזירה true אמ"ם קיימים במערך vec **לפחות** שני איברים a ו- b שונים שעבורם $a - b = c$.

סעיף ב (10 נקודות):

שנו בדף התשובות את הפונקציה שבסעיף א' כך שתחזיר true אמ"ם קיימים במערך vec **בדיוק** שני איברים כאלה.

שאלה 2 (10 נקודות)

עיינו בשיטה הבאה:

```
int foo(int a, int b) {

  if (a > 3)
    return 2 + foo(b - 1, a + 1);

  if (b <= 4)
    return 1 + foo(a - 1, b + 1);

  return 0;
}
```

ענו בדף התשובות לכל אחת מהקריאות הבאות האם השיטה תעצור או לא, ואם כן, מה היא תחזיר:

א. foo(3, 4) **(5 נקודות)**
ב. foo(4, 5) **(5 נקודות)**

שאלה 3 (25 נקודות)

פירוק לגורמים ראשוניים של מספר טבעי $n \geq 2$ הוא קבוצה לא ריקה של מספרים ראשוניים שהמכפלה שלהם שווה ל- n. למשל הפירוק של 17 הוא { 17 } והפירוק של 18 הוא { 2, 3, 3 }.

הפירוק של מספר נתון $n \geq 2$ יכול להיעשות בצורה רקורסיבית באופן הבא:

- אם n ראשוני החזר את הקבוצה { n }.
- אחרת מצא מחלק כלשהו d של n והחזר את האיחוד של הפירוק של d והפירוק של n/d.

למשל, הפירוק של 17 הוא { 17 } כי 17 ראשוני, בעוד שהפירוק של 18 הוא האיחוד של הפירוק של 2 ({ 2 }) עם הפירוק של 9 ({ 3, 3 }).

עליכם להשתמש בפונקציה getDivisor(int n) המחזירה מחלק כלשהו d של n או את n עצמו במידה ו-n ראשוני.

השלימו בדף התשובות את הפונקציה

```
int[] primeFactors(int n)
```

שמחזירה את הפירוק של n.

הניחו ש- $n \geq 2$.

שאלה 4 (45 נקודות)

Young-Tableau (Young הוא השם של הממציא) היא מטריצה n × m, שכל אחת משורותיה (בנפרד) ממוינות משמאל לימין בסדר עולה (לא בהכרח עולה ממש), ושכל אחת מעמודותיה (בנפרד) ממוינות מלמעלה למטה בסדר עולה (לא בהכרח עולה ממש). חלק מהערכים של המטריצה יכולים להיות ∞ (נתייחס אליהם כאל מקומות ריקים) . להלן **דוגמה** של Young-Tableau בגודל 4 × 3:

0	3	7	∞
7	7	∞	∞
9	10	∞	∞

סעיף א (15 נקודות):

<u>השלימו</u> בדף התשובות **בצורה היעילה ביותר** את הפונקציה

boolean isYTab (**double** [][] arr)

שמחזירה true אם"ם המערך arr הוא Young-Tableau.

הניחו ש- arr ≠ null ושהוא מטריצה.

שימו לב: ∞ ב- Java מוגדר ע"י הקבוע Double.POSITIVE_INFINITY.
Java מתייחס לערך בצורה טבעית, ומבצע השוואות בינו לבין ערכים אחרים בצורה המקובלת: ∞ שווה לעצמו וגדול מכל היתר.

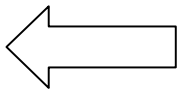
סעיף ב (15 נקודות):

הוספת איבר x ל- Young Tableau ,yTab, לא ריקה יכולה להיעשות באופן הבא:

- סמן את המשבצת הימנית התחתונה של yTab כ- current.
- חזור על התהליך הבא כדי למקם את x נכון בטבלה:
  - תתבונן בערך u שבמשבצת שמעל ל- current ובערך v שבמשבצת שמשמאל (אם ישנן משבצות כאלה)
  - אם לפחות אחד מהערכים האלה (u או v) גדול מ- x,
    - מצא מי הגדול מבין השניים.
    - העתק אותו ל- current.
    - וסמן את המשבצת שלו כ- current החדש.
  - אחרת סיים.
- הכנס את x ל- current.

לדוגמה, נניח שמכניסים את הערך 2 לטבלה שמימין. החצים הקטנים מסמנים מסלול העתקות אפשרי (אם כי לא יחידי) במהלך הריצה. הטבלה משמאל היא הטבלה המתקבלת לבסוף.

0	3	7	∞
2	7	7	∞
9	10	∞	∞



0	3	7	∞
7	7	∞	∞
9	10	∞	∞

השלימו בדף התשובות את הפונקציה

```
void insert(double[][] yTab, double x)
```

שמוסיפה את x ל- Young Tableau, yTab.

הניחו ש- yTab ≠ null, ושהיא מטריצה מסוג Young Tableau.

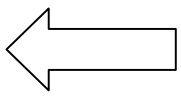
סעיף ג (15 נקודות):

שינוי של ערך קיים לערך אחר גדול ממנו x במקום נתון current ב- Young Tableau יכול להיעשות באופן הבא:

- התבונן בערך d שבמשבצת מתחת ל- current ובערך e שבמשבצת שמימין (אם ישנן משבצות כאלה כלל ועיקר).
- אם לפחות אחד מהם (d או e) קטן מ- x.
  - מצא מי הקטן מביניהם.
  - העתק אותו ל- current.
  - סמן את המשבצת שלו כ- current החדש.
  - **ובצורה רקורסיבית** שנה את הערך של ה- current החדש ל- x.
- אחרת הכנס את x ל- current.

לדוגמה, נניח שמשנים את הערך של המשבצת השמאלית העליונה מ- 0 ל- 9. החצים הקטנים מסמנים מסלולו העתקות אפשרי (אם כי לא יחידי) במהלך הריצה. הטבלה משמאל היא הטבלה המתקבלת לבסוף.

3	7	7	∞
7	9	∞	∞
9	10	∞	∞



0	3	7	∞
7	7	∞	∞
9	10	∞	∞

השלימו בדף התשובות את השיטה

```
void changeVal(double[][] yTab, int i, int j, double newVal)
```

שמשנה את הערך של המטריצה מסוג Young Tableau, yTab, במשבצת yTab[i][j], ל- newVal.

הניחו ש- yTab ≠ null, ושהיא מטריצה מסוג Young Tableau.

מספר נבחן: _____

מבוא למדעי המחשב
202-1-101-1

סמסטר ב', תשס"ז, 2007/8
בוחן

דף תשובות

שאלה 1 סעיף א (10 נקודות)

```
private static boolean check (int [] vec, int c) {

  boolean exist = false;

   for (int i = 0 ; (i < vec.length-1) & (!exist) ; i = i+1)
          for (int j = i+1 ; (j < vec.length) & (!exist) ; j = j+1)
                 if (vec[i] –vec[j] ==c | vec[j]-vec[i] ==c)
                        exist = true;

  return exist;
}
```

שאלה 1 סעיף ב (10 נקודות)

```
private static boolean check (int [] vec, int c) {

  int count = 0;

   for (int i = 0 ;  i < vec.length-1 & !exist ; i = i+1)
          for (int j = i+1 ; (j < vec.length) & (!exist) ; j = j+1)
                 if (vec[i] –vec[j] ==c | vec[j]-vec[i] ==c)
                        count = count+1;


  return (count ==1);
}
```

שאלה 2 (10 נקודות)

א. **(5 נקודות)** השיטה תעצור ותחזיר 1

ב. **(5 נקודות)** השיטה לא תעצור ותכנס לרקורסיה אינסופית

שאלה 3 (25 נקודות)

```
private static int[] primeFactors(int n) {

 int[] ans = null;

 int d = getDivisor(n);

 if (d==n) {

  ans = new int[1];
  ans[0] = n;
 }
 else {

  int[] arr1 = primeFactors(d);
  int[] arr2 = primeFactors(n/d);
  ans = new int[arr1.length + arr2.length];
  for (int i = 0 ; i < arr1.length ; i =i+1)
                ans[i] = arr1[i];
  for (int i = 0 ; i < arr2.length ; i =i+1)
                ans[arr1.length+i] = arr2[i];

 }

 return ans;
}
```

שאלה 4 סעיף א (15 נקודות)

```
private static boolean isYTab (double [][] arr) {

  int nRows = arr.length;
  int nCols = arr[0].length;

  for (int i = 0; i < nRows; ++i)

    for (int j = 0; j < nCols; ++j)

      if (i < nRows - 1 && arr[i + 1][j] < arr[i][j] | j < nCols – 1 && arr[i][j + 1] < arr[i][j])
        return false;

  return true;

}
```

שאלה 4 סעיף ב (15 נקודות)

```
private static void insert(double[][] yTab, double x) {

  int i = yTab.length - 1;
  int j = yTab[0].length - 1;

  boolean done = (i == 0 || yTab[i - 1][j] <= x) & (j == 0 || yTab[i][j - 1] <= x);

  while (!done) {

    double un = x; // upper value
    double ln = x; // left value

    if (i > 0 && yTab[i - 1][j] > x)
      un = yTab[i - 1][j];

    if (j > 0 && yTab[i][j - 1] > x)
      ln = yTab[i][j - 1];

    if (un > x | ln > x) {

      if (un > ln) {
        yTab[i][j] = yTab[i - 1][j];
        --i;
      } else {
        yTab[i][j] = yTab[i][j - 1];
        --j;
      }

      done = (i == 0 || yTab[i - 1][j] <= x) & (j == 0 || yTab[i][j - 1] <= x);

    } else
      done = true;

  }

  yTab[i][j] = x;

}
```

שאלה 4 סעיף ג (15 נקודות)

```
void changeVal(double[][] yTab, int i, int j, double newVal) {

  int lstRow = arr.length - 1;
  int lstCol = arr[0].length - 1;

  boolean done = (i == lstRow || arr[i + 1][j] >= newVal) &
                 (j == lstCol  || arr[i][j + 1] >= newVal);

  if (!done) {

    double dn = newVal; // down value
    double rn = newVal; // right value

    if (i < lstRow && arr[i + 1][j] < newVal)
      dn = arr[i + 1][j];

    if (j < lstCol && arr[i][j + 1] < newVal)
      rn = arr[i][j + 1];

    if (dn < rn) {

      arr[i][j] = arr[i + 1][j];

      changeVal(arr, i + 1, j, newVal);

    } else {

      arr[i][j] = arr[i][j + 1];

      changeVal(arr, i, j + 1, newVal);
                          }
  } else

    arr[i][j] = newVal;

}
```