

## 1. אביב 2010 מועד ב'

<pre> 1 package braude.spring10.moedb.polymorphism; 2 public class Parent 3 { 4     int i; 5 6     public void incr( int d ) 7     { 8         if( d &gt; 0 ) { 9             this.i += d; 10            this.incr( d - 1 ); 11        } 12 13    public String toString() { 14        return( "Parent's i is: " + i ); 15    } 16 } </pre>	<pre> 6 package braude.spring10.moedb.polymorphism; 7 public class Child extends Parent 8 { 9     int i; 10 11    public void incr( int d ) 12    { 13        if( d &gt; 0 ) { 14            this.i += d; 15            super.incr( d / 2 ); 16        } 17    } 18 19    public String toString() { 20        return( super.toString() + "\nChild's i is: " + i ); 21    } 22 } </pre>
---	---

```

31 package braude.spring10.moedb.polymorphism;
32 public class Test
33 {
34     public static void main( String[] args )
35     {
36         Parent a = new Child();
37         a.incr( 10 );
38         System.out.println( a );
39     }
40 }

```

**שאלה 1:**

נתונות שלוש מחלקות: Parent, Child, היורשת מ-Parent, ו-Test המכילה רק main(). הן אינן מבצעות שום דבר בעל ערך, אבל הן מפגינות כמה מההתנהגויות המיוחדות של Java. הקוד של המחלקות נתון בעמוד הבא. למדו אותן וענו על שבע השאלות הבאות.

- א. הפקודה `this.i += d;` מופיעה פעמיים: בשורה 8 במחלקה Parent ובשורה 23 במחלקה Child. בהקשר של התכנית הנתונה ב-Test, האם השם `i` מתייחס לאותו משתנה בשני המקרים? פרטו לאיזה משתנה (או משתנים) הכוונה בכל התייחסות.
- ב. באותו ההקשר, האם השם `d` מתייחס לאותו משתנה בשני המקרים? פרטו.
- ג. ושוב באותו ההקשר, האם המחווה `this` מתייחס לאותו העצם בשני המקרים? פרטו.
- ד. מה יהיה הפלט של התכנית אם תורץ?
- ה. המתודה `incr()` מוגדרת בשורה 5 ב-Parent כ-`public`. האם יהיה הבדל אם המתודה תוגדר כ-`protected`? הבדל יכול להיות אי יכולת להתקמפל או, לחילופין, תוצאה שונה בריצה.
- ו. המתודה `incr()` מוגדרת בשורה 20 ב-Child כ-`public`. האם היה הבדל לו המתודה היתה מוגדרת `protected` (ו-`incr()` ב-Parent היתה נשארת `public`)? הגדירו הבדל כמו בסעיף ה' הקודם. (אם לא למדתם מהי התשובה, נסו להגיע אליה בדרך ההגיון.)
- ז. האם התשובות לשאלות ה' ו-ו' שלעיל ישתנו אם המחלקה Test תוגדר ב-`package` אחר? (לדוגמא, עקב מחיקת השורה 31 בהגדרתה; מחלקות Parent ו-Child ישארו ללא שינוי, ב-`package` המקורי.)

**2. קיץ 2016 מועד א'****שאלה 1** (25 נקודות)

שאלה זו בוחנת את הבנתכם בתחום הפולימורפיזם. יש בה 2 סעיפים. להלן שלוש מחלקות, A, B, ו-Main:

א. האם הקוד לעיל מתקמפל? אם לא, ציינו אילו שורות שגויות ומדוע. (השתמשו במספרי השורות משמאל לעיל כדי לציין את השורות השגויות, אם צריך.)

ב. אם מצאתם שגיאות בסעיף (א), מחקו את השורות השגויות. בהנחה שהקוד שנתר עובר הדרה ללא שגיאה, איזה פלט יופק אם יורץ?

```
public class A
{
    public void print()
    {
        System.out.println( "A" );
    }

    public void printObject( A a )
    {
        System.out.println( "Obj A" );
    }

    public static void printAll( List<A> list )
    {
        System.out.println( "Begin list:" );
        for( A a : list )
            a.print();
        System.out.println( "End list" );
    }
}
```

```
public class B extends A
{
    public void print()
    {
        System.out.println( "B" );
    }

    public void printObject( B b )
    {
        System.out.println( "Obj B" );
    }
}
```

```
1  public class Main
2  {
3      public static void main( String[] args )
4      {
5          A a = new A();
6          B b = new B();
7          LinkedList<A> alist = new LinkedList<A>();
8          alist.add( a );
9          alist.add( b );
10         LinkedList<B> blist = new LinkedList<B>();
11         blist.add( a );
12         blist.add( b );
13         ( (B) a ).print();
14         ( (A) b ).print();
15         A.printAll( alist );
16         b.printAll( blist );
17         a.printObject( b );
18         b.printObject( (B) a );
19         b.printObject( (A) b );
20     }
21 }
```

### 3. קיץ 2016 מועד א'

#### שאלה 2: (20 נקודות)

שאלה זו עוסקת בהעברת פרמטרים בין מתודות.

נתונה המחלקה הבאה:

במתודה `main()` יש 4 פקודות הדפסה. מה לדעתכם יהיה הפלט שיודפס אם מתודה זו תורץ?

```
public static void main( String[] args )
{
    int x = 1;
    int[] y = { 2, 3, 4 };
    String z = "5";
    Stack<String> s = new Stack<String>();
    s.push( "6" );
    s.push( "7" );

    func( x, y, z, s );

    z = z.replaceAll( "5", "6" );
    z.replaceAll( "6", "8" );
    System.out.println( x );
    System.out.println( y[0] );
    System.out.println( z );
    System.out.println( s.pop() );
}
```

```
public class Mystery
{
    public static void func( int x,
                             int[] y,
                             String z,
                             Stack<String> s )
    {
        x = 1111;
        y[0] = 2222;
        y = new int[5];
        y[0] = 4444;
        z = "5555";
        s.push( "9999" );
    }
}
```

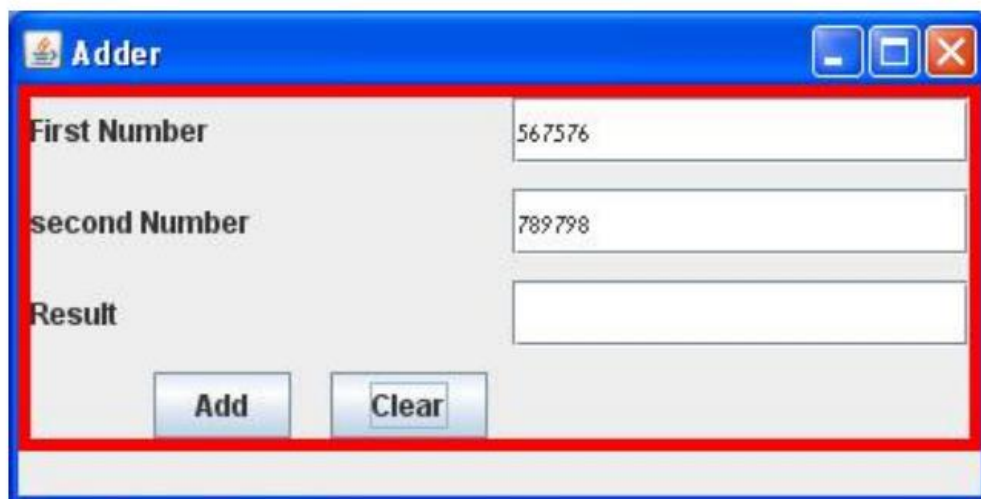
## 4. אביב 2010 מועד ב'

### שאלה 2

בשאלה זו נתון GUI והקוד המנסה לממש אותו, אלא שבתכנית נפלו מספר שגיאות ועל כן אינה פועלת. אפשר שבמימוש של תכונה יחידה נפלו מספר שגיאות, ועל כן לא יהיה די בתיקון רק אחת מהן.

התכנית היא דגם מאד פשוט של מחשבון, שיכול לבצע רק פעולה אחת: חיבור. יש שני שדות להכנסת מספרים, ושדה אחד לקבלת תוצאה. אם מכניסים מספרים לשדות הקלט ולוחצים על הכפתור add, סכומם אמור להופיע בשדה התוצאה. הכפתור clear מוחק את תוכן כל השדות, ו-exit משמש לסיום התכנית. הנה האפליקציה כפי שהיתה צריכה להיות:

כפי שהתכנית עומדת, מה שמוצג על המסך נראה כך:



בנוסף, אף אחד מהכפתורים אינו מגיב.

להלן הקוד היוצר אובייקט של מחשבון:

```
public static void main(String[] args) {  
    Calculator calculator = new Calculator();  
}
```

הקוד של המחשבון עצמו:

```
public class Calculator implements ActionListener
{
    JFrame    jf;
    JPanel    jp;
    JButton    exit;
    JButton    add;
    JButton    clear;
    JTextField first;
    JTextField second;
    JTextField result;

    public static void main( String[] args )
    {
        Calculator c = new Calculator();
    }

    public Calculator()
    {
        init();
    }

    public void init()
    {
        SwingUtilities.invokeLater( new Runnable()
        {
            public void run()
            {
                initWindow();
            }
        } );
    }

    public void actionPerformed( ActionEvent e )
    {
        if( e.getSource() == add )
        {
            double d1 = Double.parseDouble( first.getText() );
            double d2 = Double.parseDouble( second.getText() );
            double r = d1 + d2;
            result.setText( Double.toString( r ) );
        }
        else if( e.getSource() == exit )
        {
            System.exit( 1 );
        }
    }
}
```

```
public void initWindow()
{
    JPanel pb = new JPanel();
    pb.setLayout( new BoxLayout( pb, BoxLayout.X_AXIS ) );
    add = new JButton( "Add" );
    pb.add( add );
    clear = new JButton( "Clear" );
    clear.addActionListener( this );
    pb.add( clear );

    jp = new JPanel();
    jp.setBorder( BorderFactory.createLineBorder( Color.red, 5 ) );
    jp.setLayout( new GridLayout( 0, 2, 10, 10 ) );
    JLabel l1 = new JLabel( "First Number" );
    l1.setSize( 50, 10 );
    jp.add( l1 );
    first = new JTextField( 20 );
    jp.add( first );

    JLabel l2 = new JLabel( "second Number" );
    l2.setSize( 50, 10 );
    jp.add( l2 );
    second = new JTextField( 20 );
    jp.add( second );

    JLabel lr = new JLabel( "Result" );
    lr.setSize( 50, 10 );
    jp.add( lr );
    result = new JTextField( 20 );
    jp.add( result );
    jp.add( pb );

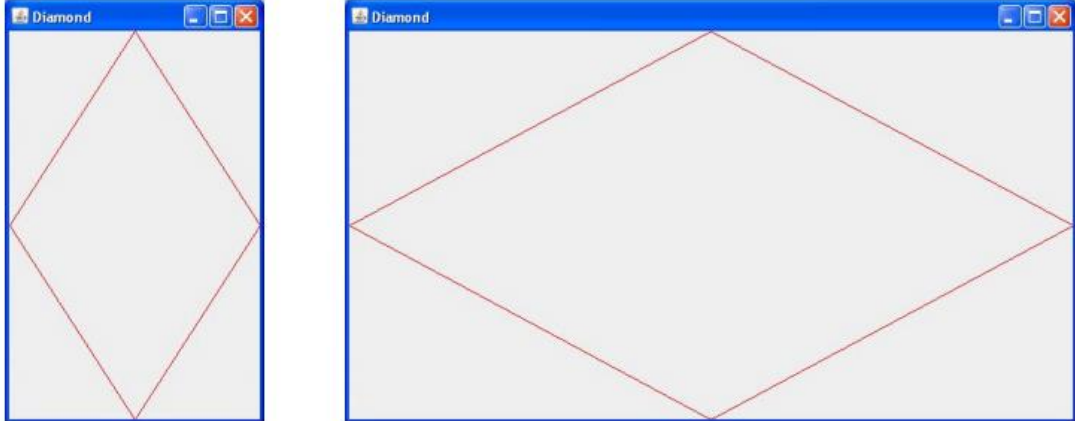
    jf = new JFrame();
    jf.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    jf.setLayout( new BoxLayout( jf.getContentPane(), BoxLayout.Y_AXIS ) );
    jf.setPreferredSize( new Dimension( 400, 200 ) );
    jf.setTitle( "Adder" );
    jf.add( jp );
    jf.pack();
    jf.setVisible( true );
}
```



## 5. סתיו 2010 מועד ב'.

**שאלה 2** (30 נקודות)

כתבו תכנית Java שלמה שפותחת חלון על המסך ובו מצויר מעויין בצבע אדום, המשיק לצידי החלון. על המעויין להשאר משיק לצידי החלון גם אם החלון משנה את ממדיו, כמתואר בצילומי המסך הבאים:



אל תשכחו להגדיר main מתאים.

יתכן ותמצאו שימוש במתודה הבאה המוגדרת במחלקה Graphics :

**drawPolygon**

```
public abstract void drawPolygon(int[] xPoints,
                                int[] yPoints,
                                int nPoints)
```

Draws a closed polygon defined by arrays of  $x$  and  $y$  coordinates. Each pair of  $(x, y)$  coordinates defines a point.

This method draws the polygon defined by  $nPoint$  line segments, where the first  $nPoint - 1$  line segments are line segments from  $(xPoints[i - 1], yPoints[i - 1])$  to  $(xPoints[i], yPoints[i])$ , for  $1 \leq i \leq nPoints$ . The figure is automatically closed by drawing a line connecting the final point to the first point, if those points are different.

**Parameters:**

- $xPoints$  - a an array of  $x$  coordinates.
- $yPoints$  - a an array of  $y$  coordinates.
- $nPoints$  - a the total number of points.

**See Also:**

[fillPolygon\(int\[\], int\[\], int\)](#), [drawPolyline\(int\[\], int\[\], int\)](#)

## 6. סתיו 2010 מועד א'

**שאלה 3** (25 נקודות)

נתון ממשק `Stack<E>` המתאר מבנה נתונים של מחסנית (LIFO):

```
package braude.stack;

public interface Stack<E>
{
    public int      size();
    public boolean  isEmpty();
    public void     clear();
    public void     push( E element );
    public E        peek();
    public E        pop();
}
```

בשאלה זו אתם מתבקשים לממש (לכתוב את הקוד) של מחלקה בשם `ArrayStack<E>` המממשת את הממשק שלעיל. על המימוש לתמוך במחסנית בגודל לא מוגבל. ההתנהגות של המתודות המוגדרות בממשק מתוארת בטבלה הבאה:

<code>public int size();</code>	מחזיר את מספר העצמים שבמחסנית.
<code>public boolean isEmpty();</code>	מחזיר <b>true</b> אם המחסנית ריקה, אחרת <b>false</b> .
<code>public void clear();</code>	מרוקן את המחסנית (מוחק את כל העצמים שבה).
<code>public void push( E element );</code>	מוסיף את העצם <code>element</code> לראש המחסנית.
<code>public E peek();</code>	מחזיר את העצם שבראש המחסנית (העצם שהוכנס אחרון) מבלי לשנות את המחסנית. מחזיר <b>null</b> אם המחסנית ריקה.
<code>public E pop();</code>	מחזיר את העצם שבראש המחסנית (העצם שהוכנס אחרון) ומסלק אותו מן המחסנית. מחזיר <b>null</b> אם המחסנית ריקה.

במימוש הנדרש אתם רשאים להשתמש במחלקה המוגדרת מראש `ArrayList<E>`, כדי לאכסן את המידע הדרוש למחסנית. להלן תיאור של תכונות של `ArrayList<E>` שיתכן שתזדקקו להן:

<code>public int size();</code>	מחזיר את מספר האברים שברשימה
<code>public boolean isEmpty();</code>	מחזיר <b>true</b> אם הרשימה ריקה, אחרת <b>false</b> .
<code>public void clear();</code>	מוחק את כל האברים שברשימה.
<code>public void add( E elem );</code>	מוסיף את העצם <code>elem</code> לרשימה.
<code>public void remove( int i );</code>	מסלק את העצם שבמקום ה- <code>i</code> מהרשימה.
<code>public Iterator&lt;E&gt; iterator();</code>	מחזיר איטרטור עבור הרשימה.



## 7. סתיו 2010 מועד א'

## שאלה 4 (25 נקודות)

בהמשך לשאלה 3 לעיל, אתם מתבקשים לקודד את המחלקה `BlockingArrayStack<E>`. מחלקה זו יורשת מ-`ArrayStack<E>`, ויודעת לבצע בדיוק את אותן פעולות, בהבדל אחד: במקום להחזיר `null` במקרה שהמחסנית ריקה, המתודות הרלוונטיות עוצרות ומחכות עד שיתווסף משהו למחסנית. הטבלה הבאה זהה לזו המתארת את `ArrayStack<E>` לעיל, פרט להבדל המודגש:

<code>public int size();</code>	מחזיר את מספר העצמים שבמחסנית.
<code>public boolean isEmpty();</code>	מחזיר <code>true</code> אם המחסנית ריקה, אחרת <code>false</code> .
<code>public void clear();</code>	מרוקן את המחסנית (מוחק את כל העצמים שבה).
<code>public void push( E element );</code>	מוסיף את העצם <code>element</code> לראש המחסנית.
<code>public E peek();</code>	מחזיר את העצם שבראש המחסנית (העצם שהוכנס אחרון) מבלי לשנות את המחסנית. <b>במקרה שכרגע המחסנית ריקה, המתודה מחכה ולא חוזרת עד שיש משהו להחזיר.</b>
<code>public E pop();</code>	מחזיר את העצם שבראש המחסנית (העצם שהוכנס אחרון) מבלי לשנות את המחסנית. <b>במקרה שכרגע המחסנית ריקה, המתודה מחכה ולא חוזרת עד שיש משהו להחזיר.</b>

מחלקה כמו `BlockingArrayQueue<E>` נחוצה רק בסביבה מרובת חוטים. אם יש רק חוט אחד פעיל, והוא זה המבקש לשלוף איבר מתוך מחסנית ריקה, אין כל טעם שיחכה כיון שאין פעילות במקום אחר שעשויה לשנות את המצב אי פעם. רק אם קיימים חוטים רבים אפשר שבשעה שחוט אחד מחכה לתשובה, חוט אחר עשוי להכניס עצם למחסנית ולשחרר את החוט הראשון מהציפיה.

כיון שיש הרבה חוטים, אפשר ששני חוטים יבצעו שתי מתודות (או את אותה המתודה) של אותו עצם בו בזמן (הניחו שיש הרבה מעבדים, אחד לכל חוט). כדאי לבדוק אילו מתודות לא טוב שיתבצעו בבת אחת, ולמנוע זאת.

מותר להסתמך על כך שקיים מימוש ל-`ArrayStack<E>` גם אם לא עניתם על השאלה הקודמת, אך נראה לי שיהיה קל יותר לענות אם יודעים מהו המימוש הזה בפרוטרוט.

בתשובות עליכם לתת מימוש רק במקומות הנחוצים. מתודות שאינכם רוצים לשנות השאירו ללא מימוש. אם ברצונכם לשנות הגדרות של מתודות קיימות, עשו זאת בגוף הטקסט.

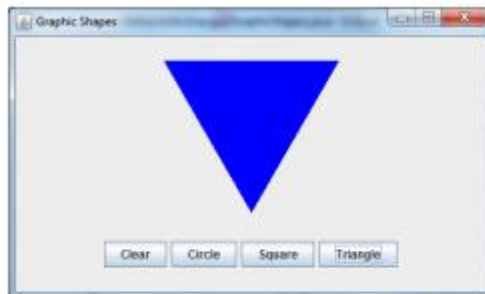
בפתרון שלי 30 שורות קוד.

## 8. סתיו 2014 מועד א'

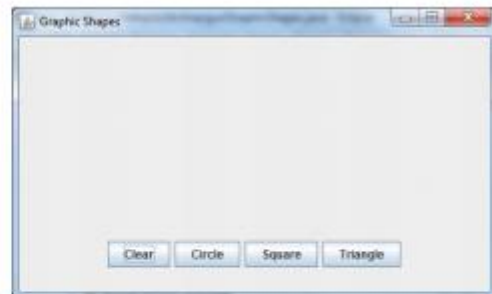
## שאלה 4: (20 נקודות)

בשאלה זו נתון GUI ותכנית המנסה לממש אותו, אלא שבתכנית נפלו שגיאות ועל כן אינה פועלת. האם תוכלו לזהות את הטעויות ולגרום לתכנית לפעול כמצופה?

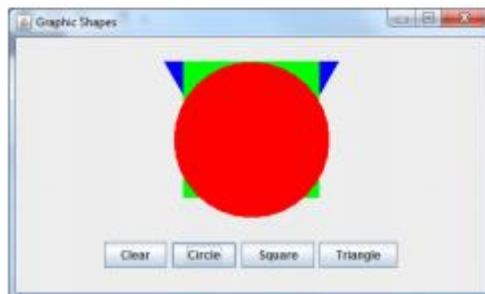
התכנית הנתונה אמורה לפתוח חלון ראשי ובו ארבעה כפתורים ושטח לציור. הכפתורים מסומנים במילים Clear (מחיקה), Circle (עיגול), Square (ריבוע), ו-Triangle (משולש). לחיצה על כפתור ה-Circle מציינת במרכז שטח הציור עיגול אדום; כפתור ה-Square מציינת ריבוע ירוק (מבלי למחוק מה שכבר מצויר); כפתור ה-Triangle מציינת משולש כחול (שוב מבלי למחוק); כפתור ה-Clear מנקה את החלון מכל הציורים. אפשר ללחוץ על הכפתורים בסדר אחר על מנת לקבל ריבוד אחר של הצורות המצוירות. להלן תמונות של ה-GUI כפי שהוא צריך לפעול:



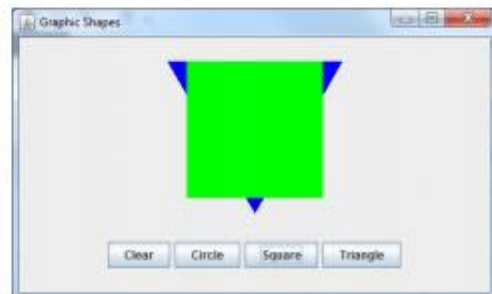
2. אחרי לחיצה על Triangle



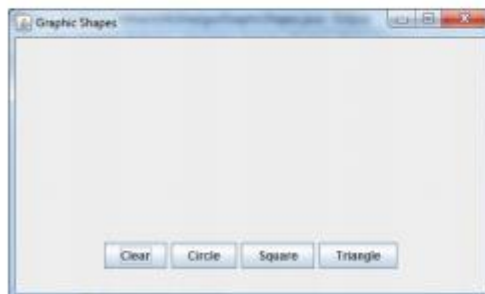
1. התחלה



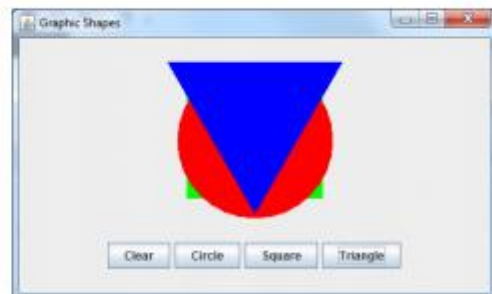
4. אחרי לחיצה על Circle



3. אחרי לחיצה על Square

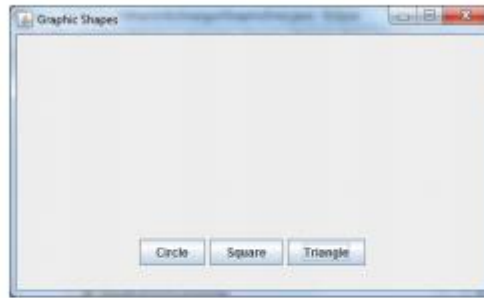


6. אחרי לחיצה על Clear



5. אחרי לחיצה נוספת על Triangle

כפי שהתכנית עומדת, היא עוברת קומפילציה ורצה ללא הודעות שגיאה. החלון שנפתח על המסך נראה כך:



לאף אחד מהכפתורים אין השפעה על הנראה על המסך.

להלן קוד התכנית. רשמו במחברת הבחינה אילו שינויים יש לבצע בקוד **באופן ברור**: באיזו מתודה, באיזו שורה, מה יש למחוק, להוסיף או לתקן. אפשר שיש בקוד יותר מטעות אחת, או גם אם מצאתם "באג", המשיכו לחפש...

```
public class GraphicError extends JPanel implements ActionListener
{
    enum        WhatToPaint { CLEAR, CIRCLE, TRIANGLE, SQUARE };

    WhatToPaint whatToPaint = WhatToPaint.CLEAR;

    JFrame      frame =      new JFrame( "Graphic Shapes" );
    JButton     circle =     new JButton( "Circle" );
    JButton     square =     new JButton( "Square" );
    JButton     triangle =   new JButton( "Triangle" );
    JButton     clear =      new JButton( "Clear" );

    public GraphicError()
    {
        frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        frame.setPreferredSize( new Dimension( 500, 300 ) );
        frame.setLayout( new FlowLayout() );
        setPreferredSize( new Dimension( 500, 200 ) );
        frame.add( this );
        frame.add( circle );
        frame.add( square );
        frame.add( triangle );
        frame.pack();
        frame.setVisible( true );
    }
}
```

```
public static void main( String[] args )
{
    GraphicError gc = new GraphicError();
}
```

```
public void actionPerformed( ActionEvent ae )
{
    if( ae.getSource() == clear )
    {
        whatToPaint = WhatToPaint.CLEAR;
    }
    else if( ae.getSource() == circle )
    {
        whatToPaint = WhatToPaint.CIRCLE;
    }
    else if( ae.getSource() == square )
    {
        whatToPaint = WhatToPaint.SQUARE;
    }
    else if( ae.getSource() == triangle )
    {
        whatToPaint = WhatToPaint.TRIANGLE;
    }
}
```

```
public void paintComponent( Graphics g )
{
    switch( whatToPaint )
    {
        case CLEAR:
            g.setColor( getBackground() );
            g.fillRect( 0, 0, getWidth(), getHeight() );
            break;

        case CIRCLE:
            g.setColor( Color.red );
            g.fillOval( 170, 20, 160, 160 );
            break;

        case SQUARE:
            g.setColor( Color.green );
            g.fillRect( 180, 20, 140, 140 );
            break;

        case TRIANGLE:
            g.setColor( Color.blue );
            int[] x = { 160, 250, 340 };
            int[] y = { 20, 176, 20 };
            g.fillPolygon( x, y, 3 );
            break;

        default:
            break;
    }
}
```

## 9. אביב 2018 מועד א'

**שאלה 5** (20 נקודות)

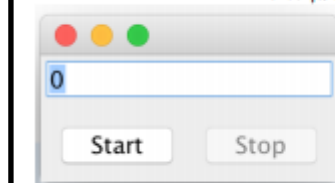
בשאלה זו, נתונה המחלקה הבאה.

```

1 public class GuiTest extends JFrame
2 {
3     private JTextField text;
4     private JButton start;
5     private JButton stop;
6
7     public GuiTest() {
8         setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
9         setLayout(new BorderLayout(5, 5)); // set frame layout,
10                                           //5 pixel gaps between components
11
12         text = new JTextField(10);
13         text.setText("0");
14
15         JPanel panel = new JPanel();
16         start = new JButton("Start");
17         stop = new JButton("Stop");
18         stop.setEnabled(false);
19         panel.add(start);
20         panel.add(stop);
21
22         add(text, BorderLayout.CENTER);
23         add(panel, BorderLayout.SOUTH);
24
25         pack();
26         setVisible(true);
27     }
28
29     public static void main(String args[]) {
30         new GuiTest();
31     }
32 }

```

הרצת ה-main הנתון יוצרת את החלון הבא:



א. (18 נק') ממשו את הפונקציות הבאה :  
 המשתמש יקליד בשדה הטקסט מספר בין 0 ל-10 (ניתן להניח כי הקלט תקין).  
 לחיצה על Start תתחיל תהליך שיחליף את המספרים בתיבת הטקסט מהמספר שהוזן ועד המספר 10 עם השהייה של 100 מילישניות בין מספר למספר.  
 בנוסף, מיד עם לחיצת הכפתור Start, הכפתור יהפוך לבלתי ניתן ללחיצה (disabled), ואילו הכפתור Stop יהפוך ללחיצ (enabled).  
 עם סיום התהליך של החלפת המספרים, ה-GUI יחזור למצב ההתחלתי.  
 בלחיצת כפתור Stop, התהליך יעצור, כפתור Start יהפוך לניתן ללחיצה (enabled), ואילו הכפתור Stop יהפוך ללא לחיצ (disabled).

ניתן לקבל את ערך תיבת הטקסט באופן הבא :  

```
int i = Integer.valueOf(text.getText());
```

  
 וניתן לעדכן את הערך באמצעות :  

```
text.setText(String.valueOf(i));
```

ניתן לשנות את הגדרת המחלקה הנתונה ולהוסיף לה דברים.  
 כמו כן, ניתן להגדיר מחלקות חדשות.  
 אל תשכחו לקשר בין הכפתורים למימוש שרשמתם.

ניתן להשתמש ב-Thread ואין צורך ב-SwingWorker (אבל ודאי שאפשרי).  
 על מנת לממש את Stop, ניתן לעשות שימוש במתודה interrupt המוגדרת במחלקה Thread או במתודה cancel המוגדרת ב-SwingWorker - לה ניתן לקרוא עם פרמטר true.

ב. (2 נק') מדוע התבקשתם לבצע את עדכון תיבת הטקסט בתהליך נפרד? מה היה קורה לו הייתם מבצעים זאת ללא תהליך נפרד?



## 10. אביב 2015 מועד ב'

**שאלה 4:** (34 נקודות)

בשאלה זו אתם מתבקשים לממש שרות הפעלת חוטים מהסוג שנותנת המחלקה `ExecutorService` שפגשנו בהרצאה. המחלקה שנכתוב, `ThreadPool`, מממשת את הממשק הבא:

בנאי: מייצר מאגר עם `thread_cnt` חוטים:  
**public ThreadPool( int thread\_cnt );**

שולח משימה לביצוע באמצעות חוט מהמאגר:  
**public void execute( Runnable r );**

מכין לסגירה: מונע קבלת משימות חדשות, אך מתיר למשימות שכבר בביצוע להסתיים:  
**public void shutdown();**

- המחלקה מגדירה שני משתנים עקריים:
1. המאגר עצמו – אוסף של חוטים.
  2. תור עבודות לביצוע – אוסף של עבודות.

תהליך העבודה הוא כדלהלן:

- הבנאי מגדיר את שני המשתנים הנ"ל; את המאגר הוא ממלא במספר מתאים של חוטים שהוא מייצר ומפעיל.
- המתודה `execute` מוסיפה משימה לביצוע בסוף התור של העבודות.
- המתודה `shutdown` מכבה דגל גלובלי שכל החוטים מסתכלים עליו.

כל אחד מהחוטים שהבנאי מייצר מריץ לולאה בה החוט לוקח משימה מראש התור ומבצע אותה, ועם סיומה חוזר ולוקח משימה נוספת, ללא הפוגה כל עוד הדגל הגלובלי דלוק. (כך `shutdown` יכולה לסגור את העסק בצורה מסודרת.) אם החוט מוצא את הדגל כבוי, הוא מסתיים.

הקושי היחיד מתעורר אם התור ריק: במקרה זה על החוט לחכות עד שתגיע משימה. אפשר לממש מתודה נוספת, שאיננה חלק מהממשק, שתפקידה להביא משימה מהתור. מתודה זו מחכה אם התור ריק, ומתעוררת כל אימת שנוספת משימה חדשה. החוט קורא למתודה זו (לחוט יש גישה גם למתודות שאינן ציבוריות).



## 11. סתיו 2010 מועד ב'

## שאלה 4 (20 נקודות)

להלן נתונה תכנית שלמה שמחקה התנהגות של תותח בועות סבון: התותח יורה בועה מידי פעם, והבועה מתקיימת לזמן מה ואז מתנדפת (או עולה השמימה, או מה שלא קורה לבועות סבון כשמגיע יומן). קצב יצירת הבועות אינו תלוי באורך חייהן, כך שאפשר מאד שמספר בועות תתקיימנה בו זמנית. במחלקה Bubble יש משתנה סטטי בשם live שגדל ב-1 בכל פעם שבוועה נולדת, וקטן ב-1 בכל פעם שבוועה הולכת לעולמה. בסוף הריצה מודפס הערך שבמשתנה live.

```
public class BubbleGun extends Thread
{
    private int count;
    private double averageDelay;
    private double averageLife;
    private Random gen;
```

```
    public BubbleGun( int count, double averageDelay, double averageLife )
    {
        this.count = count;
        this.averageDelay = averageDelay;
        this.averageLife = averageLife;
        gen = new Random( System.currentTimeMillis() );
        start();
    }

    public void run()
    {
        for( int i = 0; i < count; i++ )
        {
            new Bubble( this, getPositiveRandom( averageLife, averageLife / 5 ) );

            try {
                sleep( getPositiveRandom( averageDelay, averageDelay / 5 ) );
            }
            catch( InterruptedException e ) {}
        }
    }
```

```
    private int getPositiveRandom( double mean, double variance )
    {
        int result;
        do {
            result = (int) ( variance * gen.nextGaussian() + mean );
        }
        while( result < 0 );
        return( result );
    }

    public static void main( String[] args )
    {
        BubbleGun bg = new BubbleGun( 10000, 1, 10 );
        try
        {
            bg.join();
            Thread.sleep( 1000 );
        }
        catch( InterruptedException e ) {}

        System.out.println( Bubble.getLive() );
    }
```

```
public class Bubble extends Thread
{
    static int live = 0;
    BubbleGun bg;
    int life;

    public Bubble( BubbleGun bg, int life )
    {
        this.bg = bg;
        this.life = life;
        incLive();
        start();
    }

    private void incLive()
    {
        live++;
    }
```

```
    private void decLive()
    {
        live--;
    }

    static public int getLive()
    {
        return( live );
    }

    public void run()
    {
        try
        {
            sleep( life );
        }
        catch( InterruptedException e ) {}

        decLive();
    }
```

```
    } /* class Bubble */
```

א. (2 נקודות) מה לדעתך, על פי ההסבר לעיל, מייצג הערך של live? נמק.

ב. (6 נקודות) ומה, לדעתך, מייצג הערך של live על פי הקוד? עיין בקוד ונמק.

ג. (12 נקודות) אם המהויות שרשמת לעיל ב-א' ו-ב' אינן זהות, האם יש שינוי שאפשר לעשות בקוד שיגרום לכך שתהינה זהות? אם כן, בצע את השנוי בצורה ברורה ישירות בתוך הקוד שלהלן.

## 12. אביב 2016 מועד א'

**שאלה 4:** (30 נקודות)

בשאלה זו אתם מתבקשים לממש בשלמות שלוש מחלקות, שיחדיו מייצגות ספריה של ספרים.

1. **Book** – מחלקה זו מתארת ספר. עליה לממש את המתודות הבאות:

- a. (1) `getTitle() : String` – מחזירה את שם הספר.
- b. (1) `getAuthor() : String` – מחזירה את שם המחבר.
- c. (1) `getWidth() : double` – מחזירה את העובי של הספר (כמות המקום הוא תופס על המדף) בסנטימטרים.

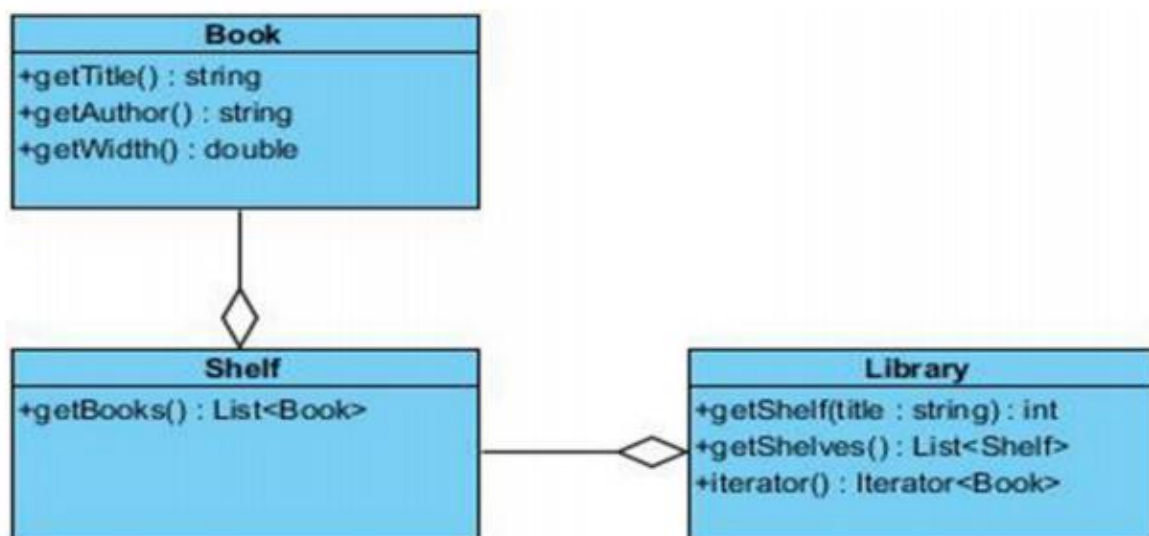
2. **Shelf** – מחלקה זו מתארת מדף. עליה לממש את המתודות הבאות:

- d. (4) `getBooks() : List<Book>` – מחזירה רשימה של כל הספרים שעל המדף.
- e. (5) `add( Book book ) : Boolean` – מוסיפה את הספר הנתון למדף, אם יש על המדף די מקום עבורו. אם לא, המתודה מחזירה `false`.

3. **Library** – מחלקה זו מתארת ספריה. עליה לממש את המתודות הבאות:

- f. (3) `getShelves() : List<Shelf>` – מחזירה רשימה של כל המדפים בספריה.
- g. (5) `getShelf( String title ) : int` – בהנתן שם של ספר, מתודה זו מחזירה את המדף עליו הספר מונח. אפשר להניח שאין בספריה שני ספרים שונים בעלי אותו השם (גם לא שני עותקים של אותו הספר).
- h. (10) `iterator() : Iterator<Book>` – מחזירה איטרטור שעובר על כל הספרים שבספריה (בלי קשר למדפים עליהם הם מונחים).

להלן דיאגרמת UML שמתארת את הנאמר לעיל:



## 13. אביב 2016 מועד א'

שאלה 1 (18 נקודות)

להלן שתי תכניות קצרות. מהו הפלט שכל אחת מהן יוצרת?

תכנית I:

```
public class TestString
{
    public static void main(String[] args)
    {
        String s1 = "bing ";
        s1 = s1.concat( "boom " );
        String s2 = "bang ";
        String s3 = "bong ";
        s3.concat( "bamm " );

        System.out.println( s1 + s2 + s3 );
    }
}
```

תכנית II :

```
public class Abcde extends Thread
{
    public synchronized void run()
    {
        try
        {
            System.out.println( "A" );
            wait();
            System.out.println( "B" );
            notify();
            System.out.println( "C" );
        }
        catch( Exception e )
        {
            System.out.println( "D" );
        }
        finally
        {
            System.out.println( "E" );
        }
    }

    public static void main(String[] args)
    {
        new Abcde().start();
    }
}
```

## 14. סתיו 2014 מועד א'

שאלה 1: התבוננו בתכנית הבאה:

```
public class A
{
    private static int j = 7;
    int i;
    int p = 10;

    public A()
    {
        i = 12;
    }

    int j()
    {
        return i + j;
    }
}

public class B extends A
{
    public B()
    {
        p = 4;
    }

    public B( int i )
    {
        this.i = j + i;
    }
}
```

```
public class C extends B
{
    int j;

    public static void main( String[] args)
    {
        A q5 = new B( 2 );
        System.out.println( q5.j() );

        A q6 = new A();
        System.out.println( q6.j() );

        C q7 = new C();
        System.out.println( q7.p );

        System.out.println( C.j );
    }
}
```

א. (5 נקודות) הקוד לעיל לא יעבור הדרה. יש בו שתי שגיאות: התוכלו לגלות אותן? ציינו אילו שורות שגויות והציעו שינוי שיגרום לקוד להתקמפל. (השגיאה איננה הגדרה של שדה ומתודה באותו השם.)

ב. (10 נקודות) אחרי שתקנתם את התכנית והיא מתקמפלת, מה יהיה הפלט הצפוי?

## 15. אביב 2018 מועד א'

ב. (6 נק') מה הפלט של הרצת תכנית ה-main הנתונה?

```

public class PassParams {
    public static void f(int x, int[] y, int[] z) {
        x = 1;
        y[0] = 2;
        z = new int[5];
        z[0] = 555;
    }
    public static void main(String[] args) {
        int x = 111;
        int[] y = { 222, 333, 444, 555 };
        int[] z = { 666, 777, 888, 999 };
        f(x, y, z);
        System.out.println(x);
        System.out.println(y[0]);
        System.out.println(z[0]);
    }
}

```

## טבלת כימוס בין מחלקות וחבילות:

	Different class but same package	Different package but subclass	Unrelated class but same module	Different module and p1 not exported
<pre> package p1; class A {     private int i;     int j;     protected int k;     public int l; } </pre>	<pre> package p1; class B { } </pre>	<pre> package p2; class C extends A { } </pre>	<pre> package p2; class D { } </pre>	<pre> package x; class E { } </pre>
	Accessible	Accessible	Inaccessible	Inaccessible