

שאלה 1, סעיף א

```
public static int sir(int n){
```

| |
|-------------------------------------|
| |
| <code>int a = 0, b = 0, c=1;</code> |
| <code>int tmp;</code> |
| <code>while (n > 1) {</code> |
| <code> tmp = a;</code> |
| <code> a = b;</code> |
| <code> b = c;</code> |
| <code> c = tmp + a + b;</code> |
| <code> n = n - 1;</code> |
| <code>}</code> |
| <code>return a;</code> |
| |
| |

```
}
```

שאלה 1, סעיף ב

```
public static int p3(int n){
```

| |
|-------------------------------------|
| |
| <code>int a = 0, b = 0, c=1;</code> |
| <code>int tmp;</code> |
| <code>while (n > 1) {</code> |
| <code> tmp = a;</code> |
| <code> a = b;</code> |
| <code> b = c;</code> |
| <code> c = tmp + a + b+1;</code> |
| <code> n = n - 1;</code> |
| <code>}</code> |
| <code>return a;</code> |
| |
| |

```
}
```


שאלה 2

אין להגדיר משתנים מסוג Pixel פרט לאלו המופיעים בתבנית.

```
public static void draw(Pixel p1, Pixel p2, int depth){
    if (depth==0){
```

| |
|---------------------------------------|
| |
| <code>Painter.drawLine(p1,p2);</code> |
| |
| |

```
} else{
```

| |
|--|
| <code>double distance = (p2.getX() - p1.getX())/3.0;</code> |
| <code>double x3 = p1.getX()+distance;</code> |
| <code>double y3 = p1.getY();</code> |
| <code>double x4 = p1.getX()+2*distance;</code> |
| <code>double y4 = p1.getY();</code> |
| |
| <code>Pixel p3 = new Pixel((int)x3, (int)y3);</code> |
| <code>Pixel p4 = new Pixel((int)x4, (int)y4);</code> |
| <code>Pixel p5 = new Pixel((int)x3,(int)(y3-distance));</code> |
| <code>Pixel p6 = new Pixel((int)x4,(int)(y4-distance));</code> |
| |
| <code>Painter.drawLine(p3, p5);</code> |
| <code>Painter.drawLine(p4, p6);</code> |
| <code>draw(p1, p3, depth-1);</code> |
| <code>draw(p5, p6, depth-1);</code> |
| <code>draw(p4, p2, depth-1);</code> |
| |

```
}
```

```
}
```

שאלה 3 סעיף א

בסעיף זה 2 פתרונות שהתקבלו.
פתרון איטרטיבי:

```
public boolean accept(Object obj){
```

```
    IntLink curr = (IntLink) obj;
```

```
    while (curr.next != null){
```

```
        if (curr.data > curr.next.data)
```

```
            return false;
```

```
            curr = curr.next;
```

```
    }
```

```
    return true;
```

```
}
```

פתרון רקורסיבי:

```
public boolean accept(Object obj){
```

```
    IntLink curr = (IntLink) obj;
```

```
    if (curr.next != null)
```

```
        return curr.data <= curr.next.data &&
```

```
            accept(curr.next);
```

```
    return true;
```

```
}
```

סעיף ב

לסעיף זה כמה פתרונות שהתקבלו. בבדיקת השאלה ניתן הדגש העיקרי למימוש נכון של Iterator – כזה שאכן מחזיר את כל האיברים העונים לקריטריון ובדיוק איברים אלו. בנוסף ניתן דגש על שימוש נכון בממשקים Filter Array. הפתרון ה"נכון":

```

public ArrayofIntLinkIterator(Array arr, int numofElements,
                               Filter filter){
    this.arr = arr;
    this.numofElements = numofElements;
    this.filter = filter;
    this.nextIndex = -1;
    for (int i=0; i < numofElements &&
          this.nextIndex == -1; i=i+1){
        if (filter.accept(arr.get(i)))
            this.nextIndex = i;
    }
}

public boolean hasNext(){
    return this.nextIndex > -1 &&
           this.nextIndex < this.numofElements;
}

public Object next(){
    if (!hasNext()) throw new NoSuchElementException();
    Object retObj = arr.get(this.nextIndex);
    this.nextIndex = this.nextIndex + 1;
    boolean found = false;
    while (this.nextIndex < this.numofElements && !found){
        if (filter.accept(arr.get(this.nextIndex)))
            found = true;
        else
            this.nextIndex = this.nextIndex + 1;
    }
    return retObj;
}
    
```

פתרון נוסף שווריאציות שונות שלו התקבלו גם כן הוא :

```
public ArrayofIntLinkIterator(Array arr, int numofElements,
                             Filter filter){
```

```
    this.arr = arr;
```

```
    this.numofElements = numofElements;
```

```
    this.filter = filter;
```

```
    this.nextIndex = 0;
```

```
}
```

```
public boolean hasNext(){
```

```
    while (this.nextIndex < this.numofElements){
```

```
        if (filter.accept(arr.get(this.nextIndex))
```

```
            return true;
```

```
        this.nextIndex = this.nextIndex + 1;
```

```
    }
```

```
    return false;
```

```
}
```

```
public Object next(){
```

```
    if (!hasNext())
```

```
        throw new NoSuchElementException();
```

```
    this.nextIndex = this.nextIndex + 1;
```

```
    return arr.get(this.nextIndex - 1);
```

```
}
```

שאלה 4, סעיף א עבור כל שיטה, ציינו בריבוע `abstract` אם הינה מופשטת. אם אינה מופשטת רשמו X בריבוע וממשו את השיטה בשורות שעומדות לרשותכם.

יש פתרונות רבים לשאלה זו. מצורפת אפשרות אחת שמגדירה שיטה מופשטת אחת.

```
public abstract boolean lessThanEqual(MyComparable other)
```

| |
|--|
| |
| |
| |

```
public X boolean lessThan(MyComparable other)
```

```
return !other.lessThanEqual(this);
```

| |
|--|
| |
| |

```
public X boolean greaterThanEqual(MyComparable other)
```

```
return other.lessThanEqual(this);
```

| |
|--|
| |
| |

```
public X boolean greaterThan(MyComparable other)
```

```
return other.lessThan(this);
```

| |
|--|
| |
| |

```
public X boolean equal(MyComparable other)
```

```
return lessThanEqual(other) && greaterThanEqual(other);
```

| |
|--|
| |
| |

```
public X boolean notEqual(MyComparable other)
```

```
return !equal(other);
```

| |
|--|
| |
| |

שאלה 4, סעיף ב

public class GalComparablePoint extends AbsComparablePoint {

```
public GalComparablePoint(int x, int y){
```

```
    super(x,y);
```

```
}
```

```
public boolean lessThanEqual(MyComparable other){
```

```
    GalComparablePoint otherPoint = (GalComparablePoint) other;
```

```
    return x < otherPoint.getX() ||
```

```
        x ==otherPoint.getX() && y =< otherPoint.getY();
```

```
}
```

```
}
```

public class LiorComparablePoint extends AbsComparablePoint {

```
public LiorComparablePoint(int x, int y){
```

```
    super(x,y);
```

```
}
```

```
public boolean lessThanEqual(MyComparable other){
```

```
    LiorComparablePoint otherPoint = (LiorComparablePoint) other;
```

```
    return x*x+y*y <
```

```
        otherPoint.getX()*otherPoint.getX() +
```

```
        otherPoint.getY()*otherPoint.getY();
```

```
}
```

```
}
```