מספר בחינה:

עמוד 1 מתוך 18 מספר סידורי: _____ מספר ת"ז: _____

בחינה בתוכנה 1

סמסטר א' תשע"ו, מועד א', 2 בפברואר ~~2015~~ 2016
ליאור וולף, ברית יונגמן, לנה דנקין

משך הבחינה שלוש שעות.

יש להניח שהקוד שמופיע במבחן הוא בגרסה java7, אלא אם כן צויין אחרת בשאלה.

יש לסמן את התשובה הטובה ביותר בתשובון. לא יינתן ניקוד על סימון תשובה בטופס הבחינה או במחברת הבחינה.

יש לנמק את התשובות המצריכות נימוק בטופס הנימוקים המצורף בלבד – נימוקים בגוף הבחינה לא יתקבלו. נימוק חסר או לא נכון עלול לגרום לאי קבלת נקודות על שאלה גם אם התשובה המסומנת היא הטובה ביותר. המקום המיועד לנימוקים בטופס מספיק לתשובות תמציתיות.

יש לצרף את טופס המבחן למחברת הבחינה. מחברת ללא טופס עזר תפסל. תשובות במחברת הבחינה לא תיבדקנה.

יש למלא מספר סידורי (מספר מחברת) ומספר ת"ז על כל דף של טופס הבחינה.

אסור השימוש בחומר עזר כלשהו, כולל מחשבונים או כל מכשיר אחר פרט לעט.

© כל הזכויות שמורות למחברים. מבלי לפגוע באמור לעיל, אין להעתיק, לצלם, להקליט, לשדר, לאחסן במאגר מידע, בכל דרך שהיא, בין מכנית ובין אלקטרונית או בכל דרך אחרת כל חלק שהוא מטופס הבחינה.

בהצלחה!

מספר בחינה:

שאלה 1 (4 נק')

התבוננו בקוד הבא והקיפו את הטענה הנכונה:

```
public final class A {
	protected int a;
	public A(int a){
		this.a = a;
		System.out.println("Created new A");
	}
	public void setA(int a){
		this.a = a;

	}
}

public class B {
	private A a;
	public B(A a){
		this.a = a;
		System.out.println("Created new B");
	}
	public static void main(String [] args){
		B b = new B(new A(3){
			public void setA(int a){
				this.a = this.a + a;
			}
		});
	}
}
```

א. יודפס:
Created new A
Created new B

ב. יודפס:
Created new B
Created new A

ג. יודפס:
Created new B

ד. שגיאת קומפילציה נימוק: מנסים ליצור מחלקה אנונימית שיורשת מהמחלקה A, אשר מוגדרת כfinal.

מספר בחינה:

שאלה 2 (4 נק')

השאלה הבאה מתייחסת לקוד הבא. התבוננו בקוד והקיפו מה יודפס למסך.

```
String s1 = new String("hello");
String s2 = new String ("hello");
String s3 = "hello";
String s4 = "hello";
System.out.print(s1.equals(s2)+" ");
System.out.print(s1.equals(s3)+" ");
System.out.print(s1 == s2 +" ");
System.out.print(s1 == s3 +" ");
System.out.print(s4 == s3);
```

א. true true false false true השוואת תוכן תיתן true, השוואת כתובת תיתן false. בנוסף s3,s4 הוגדרו כ-string literals, ולכן עבורן גם השוואת כתובת תחזיר true.

ב. true true true false true

ג. true true false false false גם תשובה זו התקבלה כנכונה.

ד. false true false false true

שאלה 3 (4 נק')

השאלה הבאה מתייחסת לקוד הבא. מה יודפס?

```
String input = " hey 1 hey 2 hey red hey blue ";
Scanner s = new Scanner(input).useDelimiter(" hey ");
while (s.hasNext())
            System.out.println(s.nextLine());
s.close();
Scanner s2 = new Scanner(input).useDelimiter(" hey ");
System.out.println(s2.nextInt());
System.out.println(s2.next());
s2.close();
```

א.

```
hey 1 hey 2 hey red hey blue
1
2
```

לא מתייחסים ל-delimiter כאשר נרצה את השורה הבאה, אלא תמיד נקבל את כל התווים עד ה-\n הבא. לעומת זאת כאשר נרצה לקבל את ה-next, הdelimiter כן משפיע.

ב.

```
1 2 red blue
1
2
```

ג.

```
hey 1 hey 2 hey red hey blue
1
hey
```

מספר בחינה:

ד.

```
1 2 red blue
1
hey
```

<u>שאלה 4 (4 נק')</u>

נתונה הפונקציה הבאה:

```
public static void func(List<String>lst){
	for (String str : lst){
		try{
			System.out.print(str.toLowerCase());
		}
		catch (NullPointerException e){
		}
		finally{
			System.out.print("!");
		}
	}
	System.out.println("\n");
}
```

איזה פלט יתקבל עבור הקריאה לפונקציה עם הרשימה הבאה?
Arrays.asList("a", null, "b")

א. תודפס המחרוזת a! ואחרי יזרק חריג מהפונקציה. במידה והקורא החיצוני לא טיפל בחריג, תודפס הודעת שגיאה.
ב. תודפס המחרוזת a!
ג. תודפס המחרוזת a!b
ד. תודפס המחרוזת a!b!
ה. תודפס המחרוזת a!!b! נימוק: מכיוון שיש try\catch, השגיאה שתיווצר עבור האיבר האחרון מטופלת והלולאה תמשיך לרוץ. בנוסף, מכיוון שיש בלוק finally, זה אומר שהתו "!" יודפס בכל איטרציה של הלולאה.

מספר בחינה:

שאלה 5 (4 נק')

התבוננו בקוד הבא: מה יקרה?

```
1.public class CurrentClass {
2.     public static void myPublicStaticMethod() {
3.           System.out.println("In myPublicStaticMethod");
4.     }
5.     private void myPrivateMethod() {
6.           System.out.println("In myPrivateMethod");
7.     }

8.     public static void main(String[] args) {
9.           myPublicStaticMethod();
10.          CurrentClass.myPrivateStaticMethod();
11.          CurrentClass currentClass = new CurrentClass();
12.          currentClass.myPrivateMethod();
13.          currentClass.myPublicStaticMethod();
14.    }
15.}
```

א. שגיאת קומפילציה בשורה 10 ניסיון לגשת למתודת מופע לפני שיצרנו מופע כלשהו. השאלה בוטלה עקב שגיאה בשם הפונקציה בשורה 10, וכל תשובה התקבלה כנכונה.

ב. יודפס:
In myPublicStaticMethod
In myPrivateMethod
In myPrivateMethod
In myPublicStaticMethod

ג. יודפס:
In myPublicStaticMethod
In myPrivateMethod
In myPublicStaticMethod

ד. שגיאת זמן ריצה

שאלה 6 (4 נק')

הקיפו את הטענה הנכונה:

א. שדות סטטים מאותחלים עם טעינת המחלקה, יש עותק יחיד בתכנית, ונגישים ממתודות סטטיות ומתודות מופע.

ב. שדות סטטים מאותחלים עם טעינת המחלקה, יש עותק יחיד בתכנית, ונגישים ממתודות סטטיות בלבד.

ג. שדות סטטים מאותחלים עם יצירת אובייקט, יש עותק יחיד בתכנית, ונגישים ממתודות סטטיות בלבד.

ד. שדות סטטים מאותחלים עם יצירת אובייקט, יש עותק עבור כל מופע, ונגישים ממתודות סטטיות ומתודות מופע.

מספר בחינה:

<u>שאלה 7 (4 נק')</u>

התבוננו בקוד הבא וענו מה יקרה

```
public interface Shape {
       public double getArea();
}

public class Square implements Shape {
      private int edge;
      public Square(int i) {
             edge = i;
      }
      public double getArea() {
             return edge*edge;
      }
}

public class Circle implements Shape {
      private int radius;
      public Circle(int i) {
             radius = i;
      }
      public int getRadius(){
             return radius;
      }
      public double getArea() {
             return Math.PI*radius*radius;
      }
}
   ....
1. public static void main(String [] args){
2.     Shape shape1 = new Square(100);
3.     Shape shape2 = new Circle(50);
4.     System.out.println( ((Circle) shape1).getRadius() );
5.     System.out.println( shape2.getRadius() );
   ....
```

א. תיזרק שגיאת זמן ריצה

ב. שגיאת קומפילציה בשורה 4

ג. שגיאת קומפילציה בשורה 5

ד. שגיאת קומפילציה בשורה 4 ובנוסף שגיאת קומפילציה בשורה 5

ה. שגיאת קומפילציה בשורה 5 ובנוסף, אם נשים את שורה 5 בהערה, תיזרק שגיאת זמן ריצה – בשורה 5 יש להוסיף casting, והרצת התכנית (ללא שורה 5) תגרור שגיאת זמן ריצה כיוון ש-shape1 הינו Square והמתודה getRadius אינה ממומשת עבור טיפוס זה.

שאלה 8 (4 נק')

כעת הניחו כי הממשק shape שונה באופן הבא: מה יקרה בהרצת התוכנית הבאה?

```
public interface Shape extends Drawable  {
       public double getArea();
}
public interface Drawable {
      public void draw();
}
….
1. Shape[] shapes = new Shape[]{new Square(10),new
                                              Circle(20),newSquare(100)};
2. for (Shape shape : shapes){
3.      System.out.println(  shape.getArea() );
4.      shape.draw();
5. }
   …
```

א. אם נשים את שורה 4 בהערה, הקוד ירוץ וידפיס את השטח של כל צורה, אחרת שגיאת קומפליציה (ללא קשר האם והמחלקות המתאימות ממשות את המתודה draw)

ב. במידה והמחלקות המתאימות ממשות את המתודה draw הקוד ירוץ, ידפיס את השטח ויקרא למתודה draw הממומשת במחלקה המתאימה.הסבר: כל מחלקה שתממש את המנשק Shape, בהכרח תאלץ לממש גם את המתודה draw, ולכן במידה וקוד המחלקות הממשות מתקמפל, גם קוד זה יתקמפל וירוץ כנדרש.

ג. במידה והמחלקות המתאימות מממשות את המתודה draw, אם נשנה את שורה 4 ל:

```
(Drawable)shape.draw();
```

הקוד ירוץ , ידפיס את השטח ויקרא למתודה draw הממומשת במחלקה המתאימה

ד. במידה והמחלקות המתאימות ממשות את המתודה draw הקוד ירוץ, ידפיס את השטח ויקרא למתודה draw הממומשת במחלקה המתאימה, אחרת תהיה שגיאת זמן ריצה בשורה 4. – תהיה שגיאת קומפילציה אם המחלקות לא מממשות את המתודה

ה. יש יותר מתשובה אחת נכונה – כיוון שיש בלבול בסוגריים ב-ג', גם תשובה זו התקבלה כנכונה.

שאלה 9 (4 נק')

לאחר מספר שכתובים של קוד קיים, נוצר מצב שבו מחלקה אבסטרקטית מכילה רק מתודות אבסטרקטיות. המתכנתת שעובדת על הקוד שוקלת להמיר את המחלקה האבסטרקטית למנשק (כלומר, במקום abstract class, להגדירה כ interface).
סמנו את הטענה הנכונה ונמקו את בחירתכם:

א. ניתן להמיר כל מחלקה אבסטרקטית שאינה מכילה מימוש במנשק והקוד יתמקפל כמו שהוא ללא שינוי אף שורת קוד באף מחלקה.

ב. ניתן להמיר כל מחלקה אבסטרקטית שאינה מכילה מימוש במנשק, אבל צריך לעדכן את הקוד של כל המחלקות היורשות.

ג. ניתן להמיר כל מחלקה אבסטרקטית שאינה מכילה מימוש בנשק, אבל בתנאי שהיא לא מכילה מתודות פרטיות.

ד. יתכן מצב בו לא נוכל להמיר מחלקה אבסטרקטית למנשק, גם אם היא מכילה רק מתודות אבסטרקטיות.
נימוק: אם המחלקה האבסטרקטית מרחיבה מחלקה אחרת, אם נמיר אותה למנשק, לא נוכל לבטא את הירושה, ובנוסף מתודות אבסטרקטיות יכולות להיות protected, ואילו כל המתודות במנשק חייבות להיות public.

<u>שאלה 10 (4 נק')</u>

התבוננו בקוד הבא ובחרו את התשובה הנכונה:

```
Collection<String> stringCollection = new HashSet<String>();
stringCollection.add(new String( "bye"));
stringCollection.add(new String( "hi"));
stringCollection.add(new String( "bye again"));
for(Iterator<String> iter = stringCollection.iterator();
   iter.hasNext();){
        String str = iter.next();
        if(str.equals("hi"))
               iter.remove();
}

for(String str: stringCollection ){
       if(str.equals("hi"))
              stringCollection.remove("hi");
}
System.out.println(stringCollection.size());
```

א. אם נחליף בסדר של שתי הלולאות, הקוד ירוץ ללא שגיאות ויודפס 2 – לא נכון, תהיה שגיאת זמן ריצה

ב. אם נחליף בסדר של שתי הלולאות, ונשתמש באופרטור == במקום equals יודפס 3 – נכון, בפרט כי ד נכון

ג. מבלי לשנות שום דבר בקוד, תיזרק שגיאת זמן ריצה – לא נכון, ירוץ ויודפס 2

ד. אם נשתמש באופרטור == במקום equals יודפס 3 ללא קשר איזו לולאה באה קודם – נכון, שום איבר לא יוסר- תשובה זו גם התקבלה כנכונה.

ה. יש יותר מתשובה אחת נכונה – זו התשובה הנכונה, כי תשובות ד ו-ב נכונות.

מספר בחינה:

<u>שאלה 11 (4 נק')</u>

בחנו את הקוד הבא והחליטו מה תוצאת ההרצה.

```
public class B extends A {
	public String someString = "B";
	public String getString(){
		return someString;
	}
}
public class A {
	public String someString = "A";
	public String getString(){
		return someString;
	}
	public static void main(String args[]) {
		A a = new A();
		A b = new B();
		B c = new B();
		System.out.println( a.someString );
		System.out.println( b.someString );
		System.out.println( c.someString );
		System.out.println( a.getString() );
		System.out.println( b.getString() );
		System.out.println( c.getString() );
	}
}
```

א. AABABB כאשר מדובר על שדות הקישור הוא סטטי, ובמתודות הוא דינמי

ב. ABBABB

ג. AABAAB

ד. ABBAAB

שאלה 12 (4 נק')

עבור הקוד הבא, בחר את הטענה הנכונה ונמקו את בחירתכם.

```
1. public class A {
2.     public void func() throws Exception{
3.         throw new Exception();
4.     }
5. }
6. public class B extends A{
7.     public void func(){
8.         super.func();
9.     }
10.    public static void main(String[] args){
11.        B b = new B();
12.        b.func();
13.    }
14. }
```

א. קיימת שגיאת קומפילציה בשורה 3.

ב. קיימת שגיאת קומפילציה בשורה 6.
ג. קיימת שגיאת קומפילציה בשורה 7.- גם תשובה זו התקבלה כנכונה, מאותו הנימוק.
ד. קיימת שגיאת קומפילציה בשורה 8. הסבר: כיוון שהמתודה של האבא זורקת שגיאה, והיא נקראת מהמתודה של הבן, על הבן להצהיר גם כי נזרקת שגיאה, או לעטוף את הקריאה עם try/catch ("הצהירי או טפלי")
ה. קיימת שגיאת קומפילציה בשורה 12.

<u>שאלה 13 (4 נק')</u>

בחנו את הקוד הבא והחליטו מהי תוצאת הקומפילציה וההרצה:

```
public class A{
	public static void main(String args[]) {
		Parent parent = new Child();
	}
}
class Parent{
	public Parent(){
		AAA();
		BBB();
	}
	private void AAA(){
		System.out.printf("AAA parent");
	}
	public void BBB(){
		System.out.println("BBB parent");
	}
}
class Child extends Parent{
	private void AAA(){
		System.out.printf("AAA Child %n");
	}
	public void BBB(){
		System.out.println("BBB Child ");
	}
}
```

א. מתקמפל ללא שגיאות ומדפיס:
AAA Parent
BBB Child נימוק: אין דריסה של מתודות פרטיות, יש עבור מתודות פומביות
ב. מתקמפל ללא שגיאות ומדפיס:
AAA Child
BBB Child
ג. מתקמפל ללא שגיאות ומדפיס:
AAA Child
BBB Parent
ד. מתקמפל ללא שגיאות ומדפיס:
AAA Parent
BBB Parent
ה. שגיאת קומפילציה – ציינו באיזו שורה יש שגיאת קומפילציה והיא הסיבה לכך

<u>שאלה 14 (4 נק')</u>

מתכנתת מימשה את המחלקה MyClass כלשהי כך שהמתודה hashCode מחזירה int רנדומי בין 0 ל 10. המימוש של equals תקין ומתחשב בכל שדות המחלקה. איזו תופעה לא יכולה להתרחש: נמקו תשובתכם.

מספר בחינה:

א. נוכל להכניס ל Set פעמיים 2 מפתחות key1 ו key2 המקיימים key.equals(k2)==true. זה אפשרי אם נכניס את המפתח בכל פעם תחת hashCode שונה, כי השימוש ב equals נעשה רק על איברים להם אותו hashCode (אחרת מבנה הנתונים היה לא יעיל כי היינו צריכים להריץ equals על כל האיברים במבנה).
תשובה זו גם התקבלה עבור נימוק שמתייחס ל-set ולא לה-hashmap (כפי שתוקן במבחן).

ב. כששולפים מפתח ששמור ב HashMap עם ערך כלשהו, נוכל לקבל תשובה שהמפתח לא קיים. – זה אפשרי כי אם ה hashCode מחזיר ערך אקראי, יתכן מצב שבו נרצה לשלוף מפתח שהכנסנו תחת hashCode אחד על פי hashCode אחר. שום אובייקט שיהיה שמור שם לא יחזיר true בהשוואה ע"י equals ולכן התשובה תהיה שהמפתח לא קיים.

ג. כששולפים מפתח ששמור ב HashMap עם ערך כלשהו, נוכל לקבל ערך לא נכון (ערך שמתאים למפתח אחר) נימוק: המימוש של equals תקין, ובשליפה נשתמש ב-equals לבדיקה

ד. שום דבר ממה שצויין. תשובה זו גם התקבלה, עם נימוק שפוסל את ג על ידי כך שהצלחנו להכניס שתי מפתחות זהים עם ערכים שונים

<u>שאלה 15 (4 נק')</u>

התבוננו בקוד הבא אשר נכתב ב Java8:

```
public class HW implements Supplier<Integer> {
  private int i;
  public Integer get() {
         return ++i;
  }
  public HW (int initialvalue){
         i = initialvalue;
  }


  public static void main(String[] args) {
    boolean selector = ??;
    if (selector){
        (Stream.generate(new HW(0))).limit(3).forEach(System.out::println);
        (Stream.generate(new HW(3))).limit(2).forEach(System.out::println);
    }
    else{
       (Stream.generate(new HW(0))).filter(s->(s<4)).forEach(System.out::println);
       (Stream.generate(new HW(3))).filter(s->(s<6)).forEach(System.out::println);
    }
  }
}
```

עבור אילו ערכים של selector יודפסו המספרים מ1..5? נמקו תשובתכם.

א. עבור אף ערך

ב. עבור TRUE בלבד נימוק:עבור ערך false, ניכנס ללולאה אינסופית.

ג. עבור FALSE בלבד

ד. עבור TRUE וגם עבור FALSE

מספר בחינה:

<u>שאלה 16 (4 נק')</u>
נתון הקוד הבא:

```
public class HelloWorld {
	private String string1 = "A";
	public String string2 = "B";
	public static String string3 = "C";
	public static void main(String[] args) {
		HelloWorld h = new HellowIsarel();
		HelloIsarel w = new HelloIsarel();
		System.out.print(h.string1);
		System.out.print(h.string2);
		System.out.print(w.string2);
		System.out.print(h.string3);
		System.out.print(w.string3);
	}
}

public class HelloIsarel extends HelloWorld {
	private String string1 = "a";
	public String string2 = "b";
	public static String string3 = "c";
}
```

סמנו את התשובה הנכונה:

א. הקוד לא מתקמפל
ב. הקוד מתקמפל ומדפיס: ABbCc השדות נקבעים על פי הטיפוס הסטטי.
ג. הקוד מתקמפל ומדפיס: AbbCc
ד. הקוד מתקמפל ומדפיס: Abbcc

מספר בחינה:

שאלה 17 (4 נק')

נתון הקוד הבא:

```
public class SimpleRunnerTest {
	public static void main(String args[]) {
		SimpleRunner r = new SimpleRunner();
		Thread t1 = new Thread(r);
		Thread t2 = new Thread(r);
		t1.start();
		t2.start();
	}
}

class SimpleRunner implements Runnable {
	int i=4;
	public void run() {
		for (int k = 0; k < 2; k++){
			i--;
			System.out.print(i + " ");
		}
	}
}
```

איזה מבין הפלטים הבאים לא ניתן לקבל בהרצת קוד זה?

א. 3 2 1 0

ב. 2 3 1 0

ג. 3 0 1 2 העליה מ-0 ל-1, ואז מ-1 ל-2 בלתי אפשרית (סדרה עולה באורך 3 אינה אפשרית)

ד. 3 1 2 0

ה. כל הפלטים יכולים להתקבל בהרצת התוכנית.

שאלה 18 (4 נק')

נרצה לממש enum באמצעות class עם בנאי פרטי, לדוגמא: את ה enum
enum workingDaysEnum {*sun*, *mon*, *tue*, *wed*, *thu*}

נממש באמצעות המחלקה:

```
public class WorkingDays{
	String day;
	public final static WorkingDays sun = new WorkingDays("sun");
	public final static WorkingDays mon = new WorkingDays("mon");
	public final static WorkingDays tue = new WorkingDays("tue");
	public final static WorkingDays wed = new WorkingDays("wed");
	public final static WorkingDays thu = new WorkingDays("thu");

	private WorkingDays(String day){
		this.day = day;
	}
	public String getDay(){
		return day;
	}
}
```

מספר בחינה:

האם מימוש זה פותר את בעיית בטיחות הטיפוסים אותה נועד לפתור ה enum? נמקו תשובתכם.

א. כן. מכיוון שיש לנו בנאי פרטי, לא נוכל לייצר כמה WorkingDays שבא לנו, אלא לעבור רק עם מה שהוגדר (בדומה ל enum). בנוסף, נוכל להגדיר שפונקציה מקבלת פרמטר מטיפוס WorkingDays וזה יבטיח שהערכים שנשלח יהיו אך ורק מתוך הקבוצה שהגדרנו, שוב: בדומה ל enum.
ב. לא – תשובה זו גם התקבלה עם הנימוק שמתייחס לעובדה כי השדה day אינו שדה פרטי.

<u>שאלה 19 (4 נק')</u>

בהמשך לשאלה 18, מה יקרה אם נוריד את המילה final מהגדרת הקבוע mon, אך שאר הקוד של WorkingDays ישאר ללא שינוי? סמנו את התשובה הנכונה ונמקו תשובתכם.

א. לא תהיה שום השפעה על השימוש במחלקה, אין צורך ב final באף אחד מהקבועים- תשובה זו גם התקבלה עם נימוק שמתייחס לכך ש-day אינו שדה פרטי.
ב. באמצעות קוד המשתמש במחלקה, נוכל לעדכן את mon כך שהביטוי
mon.getDay.equals("fri")
יהיה true
ג. באמצעות קוד המשתמש במחלקה, נוכל לעדכן את mon כך ש mon == sun יהיה true.
נימוק: נוכל לשנות כי mon=sun (כיוון ש-mon אינו final, ומוגדר public, נוכל לשנות את המצביע)
ד. באמצעות קוד המשתמש במחלקה, נוכל לייצר מצב שבו mon =! mon יחזיר true.

<u>שאלה 20 (4 נק')</u>

ע"מ לכתוב לולאת for each על אובייקט מטיפוס חדש B, B צריך:

א. לממש את המנשק Iterable.
ב. לממש את המתודות next ו hasNext.
ג. להחזיר שדה מסוג איטרטור.
ד. לא ניתן לבצע לולאת for each על אובייקט שאינו יורש מאוסף וגם שאינו מערך.

<u>שאלה 21 (4 נק')</u>

מהו היתרון של שימוש ב Comparator לעומת שימוש ב Comparable:

א. חוסך בקוד.
ב. מאפשר מיון של אותו האוסף לפי קריטריונים שונים נימוק: על ידי שימוש ב-Collection.sort, עם שליחת ה-compertor המתאים.
ג. מאפשר להשוות בין אובייקטים מטיפוסים שונים, לעומת Comparable שמאפשר השוואה רק לאובייקט מאותו הסוג.
ד. כל התשובות נכונות.

<u>שאלה 22 (4 נק')</u>

מספר בחינה:

```
public class Box<Comparable> {
	private boolean compareTo(Box b) {
		return (this.y > b.y);
	}
	double x = 0;
	double y = 0;
	public static void main(String[] args) {
		Box a = new Box();
		Box b = new Box<Integer>();
		System.out.println(a.compareTo(b));
	}
}
```

סמנו את התשובה הנכונה:

א. הקוד מתקמפל ומדפיס false. נימוק: נקראת ההשוואה בין ערכי ה-y, וכיוון שהם יהיו שווים, מוחזר false

ב. הקוד מתקמפל ומדפיס את המספר true.

ג. הקוד אינו מתקמפל. הסבירו מדוע.

ד. הקוד זורק שגיאת זמן ריצה מטיפוס ClassCastException. הסבירו מדוע.

<u>שאלה 23 (4 נק')</u>

נרצה לממש מבנה נתונים של LRUMap. מבנה נתונים נתונים זה יהווה למעשה map לכל דבר, אך גודלה של המפה יהיה מוגבל (גודל זה יועבר כפרמטר לבנאי).
כאשר נגיע לגודל המקסימלי של המפה, נמחק ממנה את המפתח אלינו ניגשנו הכי מזמן.
לדוגמא, אם גודל המפה מוגבל ל 2 איברים, הכנסנו את המפתחות "a" ולאחריו "b". בהכנסת מפתח חדש נמחק את "a" כי את "b" הכנסנו אחריו. אם לאחר הכנסת "b" שלפנו את ערכו של "a", זה אומר שנעשה בו שימוש ולכן "b" יהיה המפתח שימחק מהמפה בהכנסת מפתח חדש.
לצורך כך, עלינו להחזיר תור (Queue) שיחזיק את מפתחות המילון ויעדכן אותו כך שיהיו מסודרים לפי סדר הגישות אליהם (כל מפתח אליו ניגשנו יתווסף לסוף התור, את המחיקה נבצע מתחילת התור). שימו לב שעל מנת שמבנה הנתונים יעבוד באופן תקין, התור צריך להיות מסונכרן עם המפה באופן מלא (כלומר, כל מפתח המופיע באחד יופיע גם בשני)
המחלקה LRU תירש מ HashMap. הבנאי שלה מקבל את מספר האיברים המקסימלי שנרצה להחזיר במפה והוא ממומש באופן הבא:
לנוחותכם, מצורפים פרטי המנשק של Map ו Queue בעמוד הבא.

```
public class LRUMap<K, V> extends HashMap<K, V>{
	Queue<K> queue;
	int mapSize;
	public LRUMap(int mapSize){
		this.queue = new ArrayDeque<>();
		this.mapSize = mapSize;
	}
}
```

אילו שירותים של HashMap לא נצטרך לדרוס במחלקה LRUMap? נמקו תשובתכם.
נמקו את תשובתכם.

מספר בחינה:

א. get
ב. remove
ג. size. נימוק: מכיוון ש LRU דורס מ HashMap, הרי שניתן לרשת את size מבלי לשנות את המתודה הזו. גודל מפת ה LRU הוא כמובן גודל המפה עצמה. לגבי get ו remote – אנחנו חייבים לדרוס שתי מתודות אלה כי בשתיהן נדרש עדכון של התור queue.
ד. נצטרך לדרוס את כל המתודות האלה.

<u>שאלה 24 (4 נק')</u>
נתון המימוש של השירות put:

```
@Override
public V put(K key, V value){
        V prevVal = this.get(key);
        queue.add(key);
        this.put(key, value);
        if (queue.size() > cacheSize){
                this.remove(queue.poll());
        }
        else{
                cacheSize++;
        }
        return prevVal;
}
```

תחת ההנחה ששאר השירותים של LRU ממומשים נכון, בחרו את התשובה הנכונה ונמקו תשובתכם.

א. יתכן מצב שבו המפה תכיל יותר מ mapSize איברים.
ב. יתכן מצב שבו נצטרך למחוק מהמפה איברים בהכנסת איבר חדש למרות שהיא מכילה פחות מ mapSize איברים. נימוק: אם אנחנו מכניסים את אותו המפתח פעמיים, הוא דורס את הערך הקודם במפה אך נכנס פעמיים ל queue. כך נוצר מצב שבו ב queue יהיו יותר איברים מאשר ב mapSize. ע"פ המימוש ב put, אנחנו בודקים את הגודל של queue לפני מחיקת איבר מהמפה, ולכן יתכן מצב שבו נצטרך למחוק איברים מהמפה למרות שהיא מכילה פחות מ mapSize איברים- השאלה בוטלה עקב טעות, ולכן כל תשובה התקבלה כנכונה.
ג. במידה והוספנו באמצעות put יותר מ mapSize איברים, המפה תמיד תכיל יותר מ mapSize איברים, אלא אם כן ביצענו פעולות remove.
ד. המימוש תקין והאוסף יעבוד כנדרש.

מספר בחינה:

Method Summary for interface Queue<E>	
boolean	add(E e) Inserts the specified element into this queue if it is possible to do so immediately without violating capacity restrictions, returning true upon success and throwing an IllegalStateException if no space is currently available.
E	poll() Retrieves and removes the head of this queue, or returns null if this queue is empty.
E	remove() Retrieves and removes the head of this queue.

Method Summary for interface Map<K, V>	
boolean	containsKey(Object key) Returns true if this map contains a mapping for the specified key.
V	get(Object key) Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
V	remove(Object key) Removes the mapping for a key from this map if it is present (optional operation).
int	size() Returns the number of key-value mappings in this map.

שאלה 25 (4 נק')

הביטו בקטע הקוד הבא:

מספר בחינה:

Method Summary for interface MouseListener	
void	mouseDoubleClick(MouseEvent e)
void	mouseDown(MouseEvent e)
void	mouseUp(MouseEvent e)

```
public class MouseListenerE implements MouseListener{

	final Display d;
	final Shell s;
	public MouseListenerE() {
		d = new Display();
		s = new Shell(d);
		s.setSize(250, 200);
		s.setText("A MouseListener Example");
		s.open();
		s.addMouseListener(this);
		while (!s.isDisposed()) {
		if (!d.readAndDispatch())
				d.sleep();
		}
		d.dispose();
	}

	public void mouseDown(MouseEvent e) {
		//the code of this method is fine
		Label l = new Label(s, SWT.FLAT);
		l.setText("Mouse Button Down at:" + e.x + " " +
		l.setBounds(e.x, e.y, 150, 15);
	}

	public void mouseUp(MouseEvent e) {
	}

	public void mouseDoubleClick(MouseEvent e) {
	}

	public static void main(String[] args) {
	    new MouseListenerE();
	}
}
```

א. הקוד מתקמפל, רץ ומגיב לארועי לחיצה על עכבר. הסבר: המחלקה מממשת את כל המתודות של המנשק, ולכן זה בסדר שהיא עצמה מאזינה לאירועי עכבר.

ב. הקוד אינו מתקמפל. מדוע?

ג. הקוד מתקמפל ורץ, אך אינו מגיב לארועי עכבר.

ד. הקוד מתקמפל אך עף בזמן ריצה. הסבירו מדוע.

בהצלחה!