גרסה מספר: מספר סידורי: _____ מספר ת"ז:_____

בחינה בתוכנה 1

סמסטר א' תשע"ח, מועד א', 13 בפברואר 2018
לנה דנקין, ברית יונגמן, שי גרשטיין

משך הבחינה שלוש שעות.

סך הניקוד על השאלות בבחינה הוא 105, אך הציון המקסימלי אותו ניתן לקבל הוא 100.

יש להניח, אלא אם צויין אחרת, כי:

- הקוד שמופיע במבחן מתאים לגירסא 8 של ג'אווה.
- כל החבילות הדרושות יובאו ,ואין צורך לכתוב שורות import בגוף הקוד.
- כל מחלקה שהיא public מופיעה בקובץ java משלה.
- בכל שאלה, כל המחלקות מופיעות באותה חבילה (package).
- בזמן הבחינה, אתם נדרשים לזהות שגיאות קומפילציה שנוצרות כתוצאה מהפרת עקרונות java-יים ושימוש לא נכון במחלקות/פונקציות. במידה וישנה טעות הקלדה (סוגר חסר, שימוש באות גדולה שלא לצורך וכו') אין לראות בסיבות אלה גורמים לשגיאות קומפילציה.

.
בבחינה זו מופיע קוד שבחלקו אינו מתקמפל, אינו רץ או שנוגד את הסטנדרטים של java כפי שנלמדו בקורס, וזאת מתוך מטרה לבחון ידע והבנה של נושאים מסויימים. אין לראות בקטעי קוד אלה דוגמא לכתיבה נכונה ב java.

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

יש לנמק בתמצות את כל התשובות בטופס הנימוקים המצורף בלבד. נימוק חסר או לא נכון עלול לגרום לאי קבלת נקודות על שאלה במקרים שבהם יוחלט לקבל יותר מתשובה אחת. נימוק לתשובה אחרת מזו שסומנה על גבי טופס התשובות לא יתקבל.

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

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.

לטופס הבחינה מצורף דף לשאלות הסטודנטים. יש לכתוב שאלות שמתעוררות במהלך הבחינה בדף זה ולהעביר לסגל הקורס. שאלות ענייניות תענינה על ידי סגל הקורס בפני כל הנבחנות/ים.

© 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!

1. <u>שאלה 1:</u>
להלן שלוש טענות הקשורות לתבנית העיצוב Factory, אשר נלמדה בהקשר של שימוש במימושים של IPoint בתוך המחלקה Rectangle.
נתון המנשק ISomething ולו שני מימושים קונרקטיים XSomething ו YSomething.

<u>טענה 1:</u> שימוש ב Factory המייצר אובייקטים מטיפוס ISomething מאפשר לספק להימנע מכתיבת בנאים במחלקות הקונקרטיות הממשות את ISomething
<u>טענה 2:</u> שימוש ב Factory המייצר אובייקטים מטיפוס ISomething מאפשר ללקוח (המשתמש ב ISomething) לא להכיר את המימושים הקונקרטיים של ISomething
<u>טענה 3:</u> שימוש ב Factory המייצר אובייקטים מטיפוס ISomething מאפשר לספק להכניס שינויים במנשק ISomething, מבלי שזה ישפיע על קוד הלקוח.

בחר/י את התשובה הטובה ביותר.
א. רק טענה 1 נכונה.
ב. אף טענה אינה נכונה.
ג. רק טענה 3 נכונה.
ד. רק טענה 2 נכונה.
ה. רק טענות 3+2 נכונות.
ו. כל הטענות נכונות.
ז. רק טענות 3+1 נכונות.
ח. רק טענות 2+1 נכונות.

2. <u>שאלה 2:</u>
המחלקה A מתארת בני אדם, ו B מתארת דרקונים. נרצה לממש מחלקה C המתארת יצורים שהם גם בני אדם וגם דרקונים, כלומר, בהינתן עצם c מטיפוס C, ניתן להתייחס אליו גם כאל דרקון, וגם כאל בן אדם. הדרך הרצויה לממש את C היא ירושה גם מ A וגם מ B. מכיוון ב Java ירושה מרובה אינה אפשרית, עלינו לחפש פתרון אחר.

התבוננו בטענות הבאות:

טענה 1: ניתן להשתמש בתבנית העיצוב Bridge – מחלקה C תירש מ A ותכיל שדה מטיפוס B. זה יאפשר התנהגות זהה מכל היבט לירושה מרובה של C מהמחלקות A ו B (בפרט, נקבל התנהגות זהה אם C תירש מ B ותכיל שדה מטיפוס A).

טענה 2: ניתן להמיר את מחלקות A ו B למנשקים, כך שכל שירותי המופע יומרו לשירותי default. מחלקה C תוכל לממש את שני המנשקים A ו B, וכך למעשה נקבל התנהגות של ירושה מרובה לכל דבר.

בחר/י את התשובה הטובה ביותר:
א. שתי הטענות אינן נכונות.
ב. שתי הטענות נכונות כמו שהן.
ג. טענה 1 נכונה, טענה 2 תהיה נכונה רק אם לא קיימת מתודה בעלת חתימה זהה גם ב A וגם ב B.
ד. טענה 1 אינה נכונה, טענה 2 תהיה נכונה רק אם לא קיימת מתודה בעלת חתימה זהה גם ב A וגם ב B.

3. שאלה 3:

```
public class Box<V>{

	public void func1(Set<V> l1, Set l2){
	}

	public void func2(Set<?> l1, Set<?> l2){
	}

	public <T> void func3(Set<T> l2, Set<T> l3){
	}

	public static void main(String[] args){
		Box<String> box = new Box<>();
		Set<String> strSet = new HashSet<>();
		Set<Integer> intSet = new HashSet<>();
		box.func1(strSet, intSet); //*
		box.func2(strSet, intSet); //*
		box.func3(strSet, intSet); //*
	}
}
```

בחר/י את התשובה הטובה ביותר.

א. רק הקריאות ל func2 ו func3 מתקמפלות.
ב. רק הקריאות ל func1 ו func3 מתקמפלות.
ג. כל הקריאות המסומנות ב * אינן מתקמפלות.
ד. כל הקריאות המסומנות ב * מתקמפלות.
ה. רק הקריאה ל func1 מתקמפלת
ו. רק הקריאות ל func1 ו func2 מתקמפלות.
ז. רק הקריאה ל func2 מתקמפלת
ח. רק הקריאה ל func3 מתקמפלת

4. שאלה 4:

```
public class Test {
	public static void main(String[] args) {
		List<Integer> lst = Arrays.asList(2,3,4);
		lst.stream().
			map(x->{
				System.out.print("m ");
				return x*5;
			})
			.filter(x->{
				System.out.print("f ");
				return x%2 == 0;
			})
			.sorted()
			.peek(x->System.out.print("p "))
			.forEach(x->System.out.print(x + " "));
	}

}
```

מה יודפס בהרצת התוכנית? בחר\י את התשובה הטובה ביותר:

א. m f p m f m f p 10 20
ב. m m m f f f p 10 p 20
ג. m m m f f f p p 10 20
ד. m f m m f p 10 p 20
ה. m f p 10 m f m f p 20
ו. m f m f m f p 10 p 20
ז. m f m m f p p 10 20
ח. m f m f m f p p 10 20

5. שאלה 5:

```
try {
	//some code
}
catch{
	//some code
}
finally {
	//some code
}
```

.

בחר\י את התשובה הטובה ביותר:

א. כל התשובות מלבד תשובה זו אינן נכונות.
ב. בלוק ה finally יבוצע רק אם לא יזרק חריג בבלוק ה try.
ג. בלוק ה finally יבוצע רק אם יזרק חריג בבלוק ה try, בין אם יתפס ע"י בלוק ה catch ובין אם לאו.
ד. בלוק ה finally יבוצע רק אם יזרק חריג בבלוק ה try, ויתפס בבלוק ה catch.
ה. בלוק ה finally יבוצע רק אם יזרק חריג בבלוק ה try, ולא יתפס בבלוק ה catch.

6. <u>שאלה 6:</u>
להלן מספר טענות הקשורות למנשקים. בחר\י את הטענה הנכונה:
   א. מלבד תשובה זו, כל תשובה אחרת אינה נכונה.
   ב. מנשק יכול לרשת (extend) מנשק אחר, או מחלקה מופשטת (abstract) המכילה רק שירותים אבסטרקטיים.
   ג. מנשק אינו יכול להיות ריק – הוא חייב להכיל תוכן כלשהו (שדה\מתודה).
   ד. מלבד תשובה זו, יש לפחות שתי תשובות אחרות שהן נכונות.
   ה. מנשק יכול להכיל שדות שהם קבועים בלבד, כלומר, public final static.
   ו. החל מ Java 8 ניתן לממש כל מנשק באמצעות ביטויי למבדה (lambda).

7. <u>שאלה 7:</u>
שאלה זו מתייחסת למחלקות A, B, שמימושן אינו נתון. ניתן להניח את קיומן של מחלקות נוספות ללא שום מגבלות.

```
public void func(B b1, B b2) {
      A a1 = (A)b1;  //line 1
      A a2 = (A)b2;  //line 2
}
```

להלן מספר טענות לגבי קוד זה:

טענה 1: ניתן לממש את A ו B כך שגם שורה 1 וגם שורה 2 לא תתקמפלנה.
טענה 2: ניתן לממש את A ו B כך שהפונקציה func תתקמפל, ובזמן ריצה תיזרק שגיאת זמן ריצה בשורה 2 בלבד.
טענה 3: ניתן לממש את A ו B כך שהפונקציה func תתקמפל, ובזמן ריצה לא תיזרק שגיאת זמן ריצה.

בחר/י את התשובה הטובה ביותר.
   א. רק טענות 3+2 נכונות.
   ב. כל הטענות אינן נכונות.
   ג. רק טענה 1 נכונה.
   ד. רק טענות 2+1 נכונות.
   ה. רק טענה 2 נכונה.
   ו. רק טענות 3+1 נכונות.
   ז. כל הטענות נכונות.
   ח. רק טענה 3 נכונה.

8. שאלה 8:

```
public class A{
	public int i = 8;
}

public class B extends A{
	private String i = "9"; //*

	public static void main(String[] args) {
		A a = new B();
		System.out.print(a.i); //**
		System.out.print(((B)a).i); //***
	}
}
```

בחר\י את התשובה הטובה ביותר:

א. ישנה שגיאת קומפילציה בשורה ***
ב. הקוד מתקמפל, רץ ומדפיס 99
ג. ישנה שגיאת קומפילציה בשורה **
ד. ישנה שגיאת קומפילציה ביותר משורה אחת.
ה. ישנה שגיאת קומפילציה בשורה *
ו. הקוד מתקמפל וזורק שגיאת זמן ריצה בשורה ***.
ז. הקוד מתקמפל, רץ ומדפיס 88
ח. הקוד מתקמפל, רץ ומדפיס 89

9. שאלה 9:

שאלה זו מתייחסת לשימוש במילה השמורה static:

בחר\י את הטענה הנכונה:

א. מלבד תשובה זו, ישנן לפחות שתי תשובות נכונות.
ב. שדה אשר מוגדר כ public static מתנהג למעשה כ final גם ללא הוספת מילה זו להגדרת השדה.
ג. שירות סטטי לא יכול להיות בעל נראות private.
ד. שירות סטטי אינו ניתן לדריסה (override).
ה. שדה סטטי של המחלקה X יאותחל עם יצירתו של האובייקט הראשון מטיפוס X.
ו. במידה ומחלקה X מוגדרת כמחלקה שהיא static בתוך מחלקה <Outer<T>, המחלקה X יכולה לעשות שימוש בטיפוס הגנרי T של Outer

10. <u>שאלה 10:</u>

להלן מימוש המחלקה BoundedQueue אשר ממשת תור שגודלו חסום.

```
public class BoundedQueue<T> {
	private int capacity;
	List<T> rep  = new ArrayList<>();

	public BoundedQueue(int capacity) {
		this.capacity = capacity;
	}

	public T deq() {
		if (isEmpty()) {return null;}
		T t = rep.get(0);
		rep.remove(0);
		return t;
	}

	/*
	 * @pre t != null
	 */
	public boolean enq(T t) {
		if (rep.size() < this.capacity) {
			rep.add(t);
			return true;
		}
		return false;
	}

	public boolean isEmpty() { return rep.size() == 0;}
	/*public boolean isFull() {return rep.size() >= this.capacity;}*/
}
```

הפונק' enq תחזיר false אם התור מלא, ואחרת, תכניס לתור את האלמנט t ותחזיר true.

נרצה לעדכן את החוזה של enq בהתאם למימוש שלו, כך שישקף למשתמש את אופן השימוש הרצוי בפונקציה.

<u>הצעה 1:</u> נוציא מהערה את השירות isFull ונוסיף את התנאי האחר: @post: $ret = not isFull()

<u>הצעה 2:</u> נוסיף את תנאי האחר: @post: $ret = impl.size() < capacity

<u>הצעה 3</u>: נוציא מהערה את השירות isFull ונוסיף את תנאי הקדם: @pre: not isFull()

בחר\י את התשובה הטובה ביותר:

א. רק הצעה 1 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ב. רק הצעה 1 או 2 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ג. רק הצעה 1 או 3 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ד. כל אחת מבין ההצעות מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ה. רק הצעה 1 או 3 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ו. רק הצעה 2 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ז. רק הצעה 3 מתאימה לתאר את אופן השימוש הרצוי בפונקציה.
ח. כל אחת מבין ההצעות אינה מתאימה לתאר את אופן השימוש הרצוי בפונקציה.

11. <u>שאלה 11:</u>
שאלה זו מתייחסת למחלקה BoundedQueue אשר הוצגה בשאלה 10.

להלן שלוש הצעות לשמורות (invariants) ושמורות מימוש (implementation invariants) אשר נרצה להוסיף ל BoundedQueue: תזכורת, implies מציין גרירה לוגית (->).

<u>שמורה 1</u>: @inv: enq() == null implies isEmpty()
<u>שמורה 2</u>: @impl_inv: rep.size() <= capacity
<u>שמורה 3</u>: @impl_inv: for each t in rep , t != null

בחר\י את התשובה הטובה ביותר:
א. רק שמורה 2 מתאימה למימוש המחלקה
ב. רק שמורה 1 מתאימה למימוש המחלקה.
ג. רק שמורות 1+3 מתאימות למימוש המחלקה.
ד. רק שמורה 3 מתאימה למימוש המחלקה.
ה. רק שמורות 2+3 מתאימות למימוש המחלקה.
ו. רק שמורות 1+2 מתאימות למימוש המחלקה.
ז. כל השמורות מתאימות.
ח. כל השמורות אינן מתאימות למימוש המחלקה.

12. <u>שאלה 12:</u>
לפניכם תוכנית אשר קוראת מהקובץ f.txt. תוכנו של הקובץ הוא:

```
Sharp#c#java#sulamit
#A#B#C
```

```
public class Tester {
	private static String FILE_NAME = "f.txt";

	public static void main(String[] args) throws Exception {
		final Scanner s = new Scanner(new File(FILE_NAME));
		s.useDelimiter("#");
		String s1 = s.next();
		s.skip("#");
		String text = "";
		while (s.hasNextLine()) {
			text += s.nextLine() + ' ';
		}
		System.out.println(text);
		s.close();
	}

}
```

בהנחה שהקובץ אכן קיים בנתיב המבוקש, ואין שגיאות בזמן ריצת התוכנית, מה יהיה הפלט שלה? בחר/י את התשובה הטובה ביותר:
א. הפלט הוא java sulamit A B C
ב. הפלט הוא c#java#sulamit #A#B#C
ג. הפלט הוא java#sulamit #A#B#C
ד. הפלט הוא cjavasulamit ABC
ה. הפלט הוא javasulamit ABC
ו. הפלט הוא c java sulamit A B C

13. שאלה 13:

בחר/י את התשובה הטובה ביותר עבור התוכנית הבאה:

```
public class A {

	public static void foo() {
		System.out.println("A.foo");
	}

	/*
	 * Add this code in q14
	public void bar(Object o) {
		System.out.println("A.bar1");
	}

	public void bar(B b) {
		System.out.println("A.bar2");
	} */

}

public class B extends A {


	public static void foo() { //*
		System.out.println("B.foo()");
	}
	public static void foo(A a) {  //**
		foo();
	}
	public static void bar(A a) {
		foo(a);
	}

	public static void main(String args[]) {
		A ab = new B();
		B bb = new B();
		ab.bar(bb); //***
	}

}
```

.

א. ישנה שגיאת קומפילציה בשורה המסומנת ב **
ב. ישנה שגיאת קומפילציה בשורה המסומנת ב *
ג. התכנית מתקמפלת והפלט שלה הוא "B.foo()"
ד. התכנית מתקמפלת והפלט שלה הוא "A.foo()"
ה. ישנה שגיאת קומפילציה בשורה המסומנת ב ***

14. <u>שאלה 14:</u>

שאלה זו מתייחסת לקוד שמוצג בשאלה 13. בשאלה זו הניחו כי הקוד במחלקה A הוצא מן הערה. כעת, בחר\י את התשובה הנכונה ביותר לאחר שינוי זה.

א. ישנה שגיאת קומפילציה בשורה המסומנת ב ***
ב. התכנית מתקמפלת והפלט שלה הוא "A.bar1"
ג. התכנית מתקמפלת והפלט שלה הוא "B.foo()"
ד. ישנה שגיאת קומפילציה בשורה המסומנת ב *
ה. התכנית מתקמפלת והפלט שלה הוא "A.bar2"
ו. התכנית מתקמפלת והפלט שלה הוא "A.foo()"
ז. ישנה שגיאת קומפילציה בשורה המסומנת ב **

15. שאלה 15:

```
public class Test{
	public static final int NUMREAD = 100;
	public static final String TO = "to.txt";
	public static final String FROM = "from.txt";

	public static void main(String args[]) throws IOException {
		FileInputStream fis = new FileInputStream(new File(From));
		FileOutputStream fos = new FileOutputStream(new File(To));
		byte[] readBytes = new byte[NUMREAD];
		int numRead;
		while ((numRead = fis.read(readBytes)) != -1) {
			fos.write(readBytes, 0, NUMREAD);
		}
		fis.close();
		fos.close();
	}
}
```

תזכורת לגבי FileInputStream ו FileOutputStream:

```
public class FileInputStream
extends InputStream
```

A `FileInputStream` obtains input bytes from a file in a file system.

```
int	read(byte[] b)
```
Reads up to `b.length` bytes of data from this input stream into an array of bytes.

```
public class FileOutputStream
extends OutputStream
```

A file output stream is an output stream for writing data to a `File`

```
void	write(byte[] b, int off, int len)
```
Writes `len` bytes from the specified byte array starting at offset `off` to this file output stream.

הקובץ from מכיל 105 בתים, והקובץ to נוצר במהלך התוכנית.

הניחו כי ניתן לקרוא/לכתוב מהקבצים to ו from ללא בעיות, וכי התוכנית רצה ללא שגיאות.
בנוסף, ניתן להניח שכל התוכן שנמצא ב from נכתב ע"י התוכנית Test.

מה יקרה לאחר סיום התוכנית? בחר/י בתשובה הטובה ביותר:

א. בסיום הריצה, תוכן הקבצים זהה.
ב. בסיום הריצה, שני הקבצים מכילים אותו מספר בתים, אך התוכן שונה.
ג. בסיום הריצה, הקובץ from מכיל פחות בתים מאשר הקובץ to
ד. בסיום הריצה, הקובץ from מכיל יותר בתים מאשר הקובץ to

16. שאלה 16:

```
public class A {

	public static String str = "A"; // q17: remove static modifier

	public A() { foo(); }

	public void foo() { // q17: change to private
		System.out.println(str);
	}
}

public class B extends A {

	public static String str = "B"; // q17: remove static modifier

	public B() { foo(); }

	public void foo() { System.out.println(str); }

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

א. הקוד מתקמפל ולאחר הרצתו יודפס null B
ב. הקוד לא מתקמפל (שגיאת קומפילציה במחלקה B)
ג. הקוד מתקמפל ולאחר הרצתו יודפס null null
ד. הקוד מתקמפל ולאחר הרצתו יודפס B B
ה. הקוד מתקמפל ולאחר הרצתו יודפס A B
ו. הקוד מתקמפל ולאחר הרצתו יודפס A A

17. שאלה 17:

השאלה הבאה מתייחסת לקטע הקוד אשר הוצג בשאלה 16. כעת הניחו כי השירות foo במחלקה A מוגדר להיות בנראות private, והמשתנה str מוגדר להיות משתנה לא סטטי בשתי המחלקות A ו-B. בחר/י את הטענה הנכונה לאחר ביצוע שינויים אלו.

א. הקוד מתקמפל ולאחר הרצתו יודפס B B
ב. הקוד מתקמפל ולאחר הרצתו יודפס A A
ג. הקוד מתקמפל ולאחר הרצתו יודפס null B
ד. הקוד מתקמפל ולאחר הרצתו יודפס null null
ה. הקוד מתקמפל ולאחר הרצתו יודפס A B
ו. הקוד לא מתקמפל (שגיאת קומפילציה במחלקה B)

18. <u>שאלה 18:</u>

שאלה זו מתייחסת למחלקה A אשר מימושה נתון באופן חלקי.

```
class A{
	int i, j;

	public A(int i, int j) {
		this.i = i;
		this.j = j;
	}
	//the rest of the code is not provided
}
public class Test {
	public static void main(String[] args)
	{
		HashSet<A> hs = new HashSet<>();
		hs.add(new A(1,2));
		hs.add(new A(1,2));
		hs.add(new A(1,1));
		hs.add(new A(2,2));
		System.out.println(hs.size());
	}

}
```

להלן מספר טענות המתייחסות למימושים השונים של המחלקה A. הניחו כי הקוד הקיים של A לא ישתנה, וניתן רק להוסיף קוד. כמו כן, הניחו כי הקוד של A הוא דטרמיניסטי, כלומר, הפונקציות שלה מחזירות ערכים שאינם אקראיים (random).

טענה 1: קיים מימוש של המחלקה A עבורו יודפס 4
טענה 2: קיים מימוש של המחלקה A עבורו יודפס 2
טענה 3: קיים מימוש של המחלקה A עבורו יודפס 1
טענה 4: קיים מימוש של המחלקה A עבורו יודפס 0.
טענה 5: בהינתן המימוש המלא של A, ניתן לקבוע בוודאות מה יודפס.

הערה: גודלו ההתחלתי של ה hashSet אינו נתון ואינו משפיע על התשובה הנכונה. לצורך הפשטות, ניתן להניח שהגודל ההתחלתי גדול מ 5.

בחר/י את התשובה הטובה ביותר:

א. רק טענה 2 לא נכונה.
ב. רק טענה 4 לא נכונה.
ג. רק טענה 3 לא נכונה.
ד. רק טענה 5 לא נכונה.
ה. רק טענה 1 לא נכונה.
ו. ישנה יותר מטענה אחת לא נכונה.

19. שאלה 19:

```
public class Test {

	public static int i = 0;

	public static void main(String[] args) {
		Test t = new Test();
		t.test();
	}

	private void test() {
		Display display = Display.getDefault();
		Shell shell = new Shell(display);
		shell.setLayout(new FillLayout(SWT.VERTICAL));

		ButtonHandler bh = new ButtonHandler();
		Button b1 = new Button(shell, SWT.PUSH);
		b1.setText(""+i); //*
		b1.addSelectionListener(bh);
		Button b2 = new Button(shell, SWT.PUSH);
		b2.setText(""+i);
		b2.addSelectionListener(bh);
		Button b3 = new Button(shell, SWT.PUSH);
		b3.setText(""+i);
		b3.addSelectionListener(bh);

		//standard SWT code, shell pack, open and event loop.
	}

	public static class ButtonHandler extends SelectionAdapter {
		public void widgetSelected(SelectionEvent e) {
			if (e.getSource() instanceof Button) {
				Button b = (Button) e.getSource();
				i++; //**
				b.setText(""+i);
				}
		}

	}
}
```

בשאלה זו, הקוד מתקמפל ורץ. הניחו כי המשתמשת לחצה על הכפתור הראשון (b1) ולאחר מכן על הכפתור השני (b2). מה יקרה לאחר ביצוע פעולות אלה? בחר/י את התשובה הטובה יותר.

א. על שני הכפתורים הראשונים יופיע המספר 2 ועל השלישי המספר 0
ב. על שלושת הכפתורים יופיע המספר 1
ג. על שלושת הכפתורים יופיע המספר 2
ד. על שלושת הכפתורים יופיע המספר 0
ה. על הכפתור הראשון יופיע המספר 1, על הכפתור השני המספר 2 ועל השלישי 0
ו. על הכפתור הראשון יופיע המספר 2, ועל שני האחרים המספר 0

20. שאלה 20:

כעת הניחו כי המשתנה i מוגדר להיות שדה מופע ולא משתנה סטטי. כיצד תשתנה תשובתך? הניחו כי אם הקוד מתקמפל, המשתמשת מבצעת את אותן הפעולות כמו בשאלה הקודמת.

א. הקוד לא יתקמפל – שגיאת קומפילציה בשורה המסומנת ב-**
ב. הקוד לא יתקמפל – שגיאת קומפילציה בשורה המסומנת ב-*
ג. התשובה לא תשתנה ותישאר זהה לשאלה הקודמת
ד. הקוד יתקמפל אך התשובה תהיה שונה מהשאלה הקודמת.
ה. הקוד יתקמפל אך יזרוק שגיאת זמן ריצה לאחר הלחיצה הראשונה על הכפתור

21. שאלה 21:

```
public class A {
	public int i;

	public A(int i) {this.i = i;}
	public String toString(){return "A: " + this.i;}
}

public class Test{
	public static void main(String[] args){
		List<A> lst = …
		Collections.sort(lst);
	}
}
```

הסטודנטית שי רוצה למיין את הרשימה lst בסדר יורד על פי ערכי ה i שלהם. לצורך כך, היא מבצעת מיון באמצעות השירות Collections.sort. שי טוענת שהקוד, כפי שהוא, מבצע נכון את המשימה, ולאחר פעולת ה sort הרשימה ממויינת בסדר הרצוי.

להלן מספר טענות.

טענה 1: שי טועה. פתרון אפשרי לבעיה הוא שהמחלקה A תממש את המנשק Comparable, והמתודה ()compareTo תממש את הלוגיקה הרצויה.
טענה 2: שי טועה. פתרון אפשרי לבעיה הוא שהמחלקה A תממש את המנשק Iterable, והמתודה ()iterator תחזיר איטרטור אשר יממש את הלוגיקה הרצויה.
טענה 3: שי צודקת. הקוד כפי שהוא יבצע הדפסה של הרשימה על פי סדר יורד של ערכי ה i.
טענה 4: שי טועה. פתרון אפשרי לבעיה הוא לדרוס (override) את המתודות ()hashCode ו-()equals במחלקה A כך שיממשו את הלוגיקה הרצויה.

א. יש יותר מטענה אחת נכונה.
ב. רק טענה 2 נכונה.
ג. רק טענה 1 נכונה.
ד. רק טענה 4 נכונה.
ה. כל הטענות אינן נכונות.
ו. רק טענה 3 נכונה.