מספר סידורי: _____ מספר ת"ז:_____

בחינה בתוכנה 1

סמסטר א'+ב' תשע"ח, מועד ב', 16 בספטמבר 2018
לנה דנקין, ברית יונגמן, שי גרשטיין

משך הבחינה שלוש שעות.

יש להניח, אלא אם צויין אחרת, כי:
- הקוד שמופיע במבחן מתאים לגירסא Java8.
- 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 את כל התשובות בטופס הנימוקים המצורף בלבד. 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 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. שאלה 1:

```
public class A {

	private static void f1() {System.out.println("In A.f1");}
	private void f2() {f1();}
	public static void f3() {System.out.println("In A.f3");}
	protected void f4() {f3();}
}

public class B extends A {

	private static void f1() {System.out.println("In B.f1");}
	private void f2() {f1();}
	public static void f3() {System.out.println("In B.f3");}
	public void f4() {f3();}

	public static void main(String[] args) {
		A a = new B();
		a.f4();
		//a.f2(); //*
	}
}
```

בחנ/י את הקוד הבא ובחר/י בתשובה הנכונה ביותר:

א. התוכנית מדפיסה in B.f3. אם נוציא את השורה המסומנת ב-* מהערה הקוד לא יתקמפל
ב. התוכנית מדפיסה in B.f3. אם נוציא את השורה המסומנת ב-* מההערה יודפס גם in B.f1
ג. התוכנית מדפיסה in A.f3. אם נוציא את השורה המסומנת ב-* מהערה הקוד לא יתקמפל
ד. התוכנית מדפיסה in B.f3. אם נוציא את השורה המסומנת ב-* מההערה יודפס גם in A.f1
ה. התוכנית מדפיסה in A.f3. אם נוציא את השורה המסומנת ב-* מההערה יודפס גם in A.f1
ו. התוכנית מדפיסה in A.f3. אם נוציא את השורה המסומנת ב-* מההערה יודפס גם in B.f1

2. שאלה 2

להלן מספר טענות:
טענה 1: מחלקה פנימית סטטית יכולה להכיל רק מתודות סטטיות.
טענה 2: מנשק יכול להכיל מימושים של מתודות סטטיות ומימושים של מתודות שאינן סטטיות
טענה 3: מחלקה אבסטרקטית יכולה להכיל מימושים של מתודות סטטיות ומתודות שאינן סטטיות
בחר/י בתשובה הטובה ביותר:

א. ישנה רק טענה אחת נכונה
ב. ישנן בדיוק שתי טענות נכונות
ג. כל הטענות נכונות.
ד. כל הטענות אינן נכונות

3. שאלה 3

```
public class A {
     protected void foo(int i, double j) throws IOException{};
     private int foo(int i, int j) throws Exception {return i+j;};
}

/*
//Option 1
public class B extends A {
     protected void foo(int i, int j) throws IOException{};
     private int foo(int i, int j) throws IOException {return i+j;};
}*/

/*
//Option 2
public class B extends A {
     public void foo(int i, double j) throws IOException{};
     private int foo(int i, int j) {return i+j;};
}*/

/*
//Option 3
public class B extends A {
     private void foo(double i, double j) throws IOException{};
     private int foo(double i, int j) throws Exception {return (int)i+j;};
}*/
```

לפניכם נתון המימוש של מחלקה A, ושלוש אופציות למימוש של מחלקה B.
בחר/י בתשובה הטובה ביותר. הטענות מתייחסות לכך שנוציא מההערה כל אחת מהאופציות למימוש B בנפרד.

א. המחלקה A לא מתקמפלת ללא קשר למימושים השונים למחלקה B.
ב. המחלקה A מתקמפלת. קיימת אופציה אחת בלבד למימוש B שתתקמפל אם נוציא אותה מההערה.
ג. המחלקה A מתקמפלת, קיימות שתי אופציות בלבד למימוש B שכל אחת מהן תתקמפל אם נוציא אותה מההערה.
ד. המחלקה A מתקמפלת. כל אחת מהאופציות למימוש B תתקמפל אם נוציא אותה מההערה.
ה. המחלקה A מתקמפלת, כל אחת מהאופציות למימוש B לא תתקמפל אם נוציא אותה מההערה

בשאלה זו נפלה טעות בניסוח של אחת הטענות. מכיוון שטעות זו לא התגלתה בזמן ולא תוקנה על הלוח שאלה זו נפסלת ונקבל את כל התשובות שסומנו כנכונות.

4. <u>שאלה 4:</u>

להלן שתי מחלקות A ו B. המימושים של A ו B אינם נתונים, אך ב A ידוע שכל המתודות שמופיעות בה הן אבסטרקטיות (abstract). המימושים של I ו J לא נתונים.
ניתן להניח כי אין עוד מנשקים ו\או מחלקות אשר מממשים\יורשים מהמחלקות והמנשקים הנ"ל.
כמו כן, ניתן להניח כי כרגע התכנית מתקמפלת ורצה.

```
public interface I{ /* some code here */ }

public interface J{ /* some code here */ }

public abstract class A implements I{
     /* some code here, methods are only abstract */
}
public class B extends A implements J{
     /* some code here*/
}
```

<u>טענה 1</u>: התוכנית תפעל בדיוק באותו האופן אם נכניס את השינויים הבאים:
(i) מחלקה A תהפוך להיות מנשק אשר מרחיב (extends) את המנשק I ותתאים את הקוד שלה כנדרש.
(ii) מחלקה B תממש את שני המנשקים I ו-J במקום לרשת מ A.
(iii) שאר הקוד ללא שינוי.

<u>טענה 2</u>: התוכנית תפעל בדיוק באותו האופן אם נכניס את השינויים הבאים:
(i) המנשק J יהפוך להיות מחלקה אבסטרקטית שיורשת ממחלקה A, והקוד של J יותאם כנדרש.
(ii) המחלקה B תירש מ J במקום מ A.
(iii) שאר הקוד ללא שינוי.

א. טענה 1 תמיד לא נכונה, ישנם מקרים בהם טענה 2 נכונה
ב. שתי הטענות תמיד לא נכונות
ג. שתי הטענות תמיד נכונות
ד. ישנם מקרים בהם טענה 1 נכונה, טענה 2 תמיד לא נכונה
ה. ישנם מקרים בהם שתי הטענות נכונות, אך לא תמיד
ו. טענה 1 תמיד לא נכונה, טענה 2 תמיד נכונה
ז. ישנם מקרים בהן טענה 1 נכונה, טענה 2 תמיד נכונה
ח. טענה 1 תמיד נכונה, ישנם מקרים בהם טענה 2 נכונה

5. שאלה 5:

```
public class A {
	public static void main(String[] args) {
		Stream<Integer> s= Stream.generate(new RandomNumbers());
		System.out.println(s.filter(x-> x%3 == 0)
					.peek(x->System.out.print(x + "_"))
					.map(x->x+1)
					.peek(x->System.out.print(x + "_"))
					.anyMatch(x-> x% 10==0)
					);
	}

}
public class RandomNumbers implements Supplier<Integer>{
	Random random = new Random();
	@Override
	public Integer get() {return random.nextInt(10)+1;}
}
```

המתודה nextInt(bound) מחזירה ערך אקראי בין 0 (כולל) לבין bound (לא כולל).

מכיוון שיש שימוש בערכים אקראיים, יכולים להיות הבדלים בפלטים בין ריצות שונות. להלן אופציות לפלטים שאפשר לקבל בהרצת הקוד:

אופציה 1: `3_4_9_10_true`
אופציה 2: `3_9_4_10_true`
אופציה 3: `true`
אופציה 4: `3_4_false`
אופציה 5: `1_4_9_10_true`
אופציה 6: false

בחר/י בתשובה הטובה ביותר:

א. מבין כל האופציות, יש שתי אופציות בדיוק שהן פלטים אפשריים לריצה.
ב. מבין כל האופציות, יש שלוש אופציות בדיוק שהן פלטים אפשריים לריצה.
ג. מבין כל האופציות, יש אופציה אחת בדיוק שהיא פלט אפשרי לריצה.
ד. מבין כל האופציות, יש ארבע אופציות בדיוק שהן פלטים אפשריים לריצה.
ה. מבין כל האופציות, יש חמש אופציות בדיוק שהן פלטים אפשריים לריצה.

6. שאלה 6:

להלן מספר טענות הקשורות למחלקה java.util.Scanner.

טענה 1: ניתן להשתמש ב Scanner רק במקרים של קריאה מקובץ או במקרים של קריאה מהקלט שהוקלד ע"י המשתמש (System.in).

טענה 2: באמצעות scanner ניתן לקרוא טיפוסים פרימיטיביים כמו int בצורה ישירה, ללא המרה מפורשת ממחרוזת.

טענה 3: הפלט של המתודה nextLine לא תלוי ב delimiter שיכול להיקבע ע"י שימוש במתודה useDelimiter.

בחר/י בתשובה הטובה ביותר.

א. כל הטענות לא נכונות.
ב. רק טענה 1 נכונה.
ג. רק טענה 2 נכונה.
ד. רק טענה 3 נכונה.
ה. רק טענות 2+1 נכונות.
ו. רק טענות 3+1 נכונות.
ז. רק טענות 3+2 נכונות.
ח. כל הטענות נכונות.

7. שאלה 7:

```
public class A {
     public static void main(String[] args){
          int[][] arr1 = new int[3][3];
          int[][] arr2 = new int[3][3];
          arr1[0] = new int[] {1,2,3};
          arr2[0] = arr1[0];
          System.out.print((arr1[0]==arr2[0]) + " ");
          System.out.print((arr1[1]==arr2[1]) + " ");
          System.out.print((arr1[1][0]==arr2[1][0]));
     }
}
```

מה יודפס בעת הרצת התוכנית? בחר/י בתשובה הטובה ביותר:

א. יודפס false false false
ב. יודפס false false true
ג. יודפס false true false
ד. יודפס false true true
ה. יודפס true false false
ו. יודפס true false true
ז. יודפס true true false
ח. יודפס true true true

8. <u>שאלה 8:</u>

שאלות 8-12 מתייחסות למבנה הנתונים MyMap. מבנה זה הוא מפה (Map) בין מפתח לערך, ובנוסף, עוקב אחר סדר הגישה לנתונים. המתודה getLastAccessedKey מחזירה את המפתח שאליו בוצעה הגישה אחרונה, באמצעות שליפה (get) או באמצעות הכנסה (put). שימו לב, המעקב אחר סדר הגישות הוא רק עבור מפתחות אשר מופיעים במילון.

```
/* impl_inv: internalMap.size() == keysOrder.size() */

public class MyMap<K, V>{
     private Map<K, V> internalMap = new HashMap<>();
     private List<K> keysOrder = new ArrayList<>();

     public boolean put(K key, V value){
          if (key == null) return false;
          internalMap.put(key, value);
          moveKeyToTop(key);
          return true;
     }
     public Optional<V> get(K key){
          if (key != null) {
               moveKeyToTop(key);
               if (internalMap.containsKey(key)){
                    return Optional.of(internalMap.get(key));
               }
          }
          return Optional.empty();
     }

     private void moveKeyToTop(K key){
          keysOrder.remove(key);
          keysOrder.add(0, key);  //adds to the beginning of the list
     }

     public K getLastAccessedKey(){ return keysOrder.get(0); }

     public int getSize(){ return internalMap.size(); }

     public static void main(String[] args){
          MyMap<String, Integer> myMap = new MyMap<>();
          myMap.put("a", 1);
          myMap.put("b", 2);
          System.out.println(myMap.getLastAccessedKey()); // b
          myMap.put("a", 3);
          System.out.println(myMap.getSize());  //2
          System.out.println(myMap.getLastAccessedKey()); //a
          System.out.println(myMap.get("a").get()); //3
     }
}
```

להלן שתי טענות המתייחסות למתודה get.
טענה 1: המתודה get עשויה לזרוק חריג. ניתן להוסיף תנאי pre על מנת למנוע מצב זה, וזאת מבלי לשנות את מימוש המחלקה.
טענה 2: המתודה get מפרה (violates) את משתמר הייצוג (impl. Invariant) של המחלקה.

בחר\י בתשובה הטובה ביותר:

א. רק טענה 1 נכונה.
ב. רק טענה 2 נכונה.
ג. שתי הטענות נכונות.
ד. שתי הטענות לא נכונות.

9. <u>שאלה 9:</u>

החל משאלה זו, הניחו כי במידה והמתודה get זרקה חריג או הפרה את משתמר הייצוג, מימוש המתודה תוקן וכעת עובדת כנדרש.
נרצה לבחון מימושים אלטרנטיביים למימוש מתודה moveKeyToTop. המימוש המקורי, שהוא תקין, מופיע תחת אופציה 1.

```
//option 1 - the original code
private void moveKeyToTop(K key){
     this.keysOrder.remove(key);
     this.keysOrder.add(0, key);
}

/*
//option 2
private void moveKeyToTop(K key){
     this.keysOrder.add(0, key);
}
*/

/*
//option 3
private void moveKeyToTop(K key){
     this.keysOrder.remove(key);
     this.keysOrder.add(key); //adds to the end of the list
}
*/
```

טענה 1: המימוש באופציה 2 לא מקיים את משתמר הייצוג.
טענה 2: המימוש באופציה 3 לא מקיים את משתמר הייצוג.
טענה 3: שימוש במימוש של אופציה 3 יגרום לכך שהרצת הפונק' main תדפיס ערכים לא נכונים (הערכים הנכונים מופיעים בהערה ליד כל שורת הדפסה)
בחר/י בתשובה הטובה ביותר:

א. רק טענה 1 נכונה
ב. רק טענה 2 נכונה
ג. רק טענה 3 נכונה
ד. רק טענות 1+2 נכונות
ה. רק טענות 1+3 נכונות
ו. רק טענות 2+3 נכונות.
ז. כל הטענות נכונות.
ח. כל הטענות לא נכונות.

10. שאלה 10

נרצה להרחיב את MyMap כך שניתן יהיה לעבור על הערכים (values) לפי סדר הגישה אליהם בלולאת forEach. סדר המעבר יהיה כך שהאיבר הראשון שיחזור יהיה האיבר שמתאים למפתח האחרון שניגשנו אליו, כלומר, האיבר שהקריאה ל getLastAccessedKey תחזיר.
נעדכן את המימוש של MyMap באופן הבא:

```
public class MyMap<K, V> implements Iterable<V>{
     //the rest of the code unchanged.

     //new code here!
     @Override
     public Iterator<V> iterator() {
          return new Iterator<V>() {
               Iterator<K> internalIt = keysOrder.iterator(); // *
               public boolean hasNext() {
                    return internalIt.hasNext();
               }

               @Override
               public V next() {
                    return internalMap.get(internalIt.next()); // **
               }
          };
     }
}
```

בנוסף, נעדכן את המתודה main כך שבסופה נעבור בלולאת forEach על myMap. הקוד החדש צריך להדפיס 3 ואז 2, זאת כיוון שהגישה למפתח "a" שמתאים לערך 3 נעשתה אחרי הגישה למפתח "b" שמתאים לערך 2.

```
     public static void main(String[] args) {
          MyMap<String, Integer> myMap = new MyMap<>();
          myMap.put("a", 1);
          myMap.put("b", 2);
          System.out.println(myMap.getLastAccessedKey());
          myMap.put("a", 3);
          System.out.println(myMap.getSize());
          System.out.println(myMap.getLastAccessedKey());
          System.out.println(myMap.get("a").get());
          //new code
          //this should print 3 2
          for (Integer value : myMap) {                    //***
               System.out.print(value + " ");
          }
     }
```

בחר/י בתשובה הטובה ביותר.

א. הקוד מתקמפל ועובד כנדרש.
ב. הקוד לא מתקמפל בשורות המסומנות ב * וב **
ג. הקוד מתקמפל אך יזרוק שגיאת NullPointerException בזמן הריצה.
ד. הקוד לא מתקמפל בשורה ***.
ה. הקוד מתקמפל, אז בעת ההרצה שלו main הלולאה תדפיס יותר משני מספרים.

11. שאלה 11

במקום להפוך את המחלקה MyMap ל Iterable, כפי שעשינו בשאלה 16, נוסיף ל MyMap שירות אשר מחזיר את סידור המפתחות לפי סדר הגישה אליהם:

```
public class MyMap<K, V> {
     //the rest of the code unchanged.

     public List<K> getKeysAccessOrder(){
          return keysOrder;
     }
}
```

לפניכם מספר טענות:

טענה 1: השירות החדש יאפשר למשתמש לכתוב קוד אשר יכול לעבור בלולאה על הערכים ב MyMap לפי סדר הגישה אליהם.

טענה 2: השירות החדש יאפשר הפרה של משתמר המימוש.

טענה 3: השירות החדש יאפשר למתודה get להחזיר ערכים שלא הוכנסו דרך המתודה put.

בחר/י בתשובה הטובה ביותר:

א. רק טענה 1 נכונה
ב. רק טענה 2 נכונה
ג. רק טענה 3 נכונה
ד. רק טענות 1+2 נכונות
ה. רק טענות 1+3 נכונות
ו. רק טענות 2+3 נכונות.
ז. כל הטענות נכונות.
ח. כל הטענות לא נכונות.

12. שאלה 12

נרצה לממש את המחלקה MyCompMap שמתנהגת כמו MyMap, אך מחייבת את הערכים שלה להיות Comparable.

המחלקה MyCompMap צריכה להיות ממומשת כך שהשורה הראשונה בקוד המצורף תתקמפל, והשניה לא (Integer הוא Comparable ו Object לא):

```
MyCompMap<String, Integer> c1 = new MyCompMap<>(); //this should compile
MyCompMap<String, Object> c2 = new MyCompMap<>();  //this shouldn't compile
```

לפניכם מספר אופציות למימוש MyCompMap:

אופציה 1:
```
class MyCompMap<K, V> extends MyMap<K, Comparable>
```
אופציה 2:
```
class MyCompMap<K, V extends Comparable> extends MyMap<K, V>
```
אופציה 3:
```
class MyCompMap<K, ? extends Comparable> extends MyMap<K, V>
```

בחר/י בתשובה הטובה ביותר:

א. רק אופציה 1 מתאימה.
ב. רק אופציה 2 מתאימה.
ג. רק אופציה 3 מתאימה.
ד. רק אופציות 1+2 מתאימות.
ה. רק אופציות 1+3 מתאימות
ו. רק אופציות 2+3 מתאימות.
ז. כל אופציות מתאימות.
ח. כל אופציות לא מתאימות.

13. שאלה 13:

```
public class House {
	public int area;
	public Room room;
	public House(int a1, int a2) {
		area = a1;
		room = new Room(a2);
	}

	public class Room {
		int area; // ***
		public Room(int h) {
			this.area = h;
		}

		public void printStr() {
			System.out.println(House.this.area + "" + area);
		}
	}

	public static void main(String[] args) {
		House h1 = new House(9,7);
		House h2 = new House(8,6);
		Room temp = h1.room;
		h1.room = h2.room; // $$$
		h2.room = temp;
		h1.room.printStr();
	}
}
```

בחר/י בתשובה הטובה ביותר:

א. יש שגיאת קומפילציה בשורה *** בלבד.
ב. יש לפחות שגיאת קומפילציה אחת, בין היתר בשורה $$$.
ג. הקוד מתקמפל אך יש שגיאת זמן ריצה.
ד. הקוד רץ ללא שגיאות, והפלט הינו 97
ה. הקוד רץ ללא שגיאות, והפלט הינו 86
ו. הקוד רץ ללא שגיאות, והפלט הינו 96
ז. הקוד רץ ללא שגיאות, והפלט הינו 87
ח. הקוד רץ ללא שגיאות, והפלט הינו 99

14. שאלה 14

אילו מבין התשובות היא הנכונה? בחר/י בתשובה הטובה ביותר:

א. התאימות אחורה מאפשרת להריץ קוד שקומפל עם Jdk8 על Jre7.
ב. בסיום ריצת המפרש (interpreter) נוצרים קבצי bytecode.
ג. הקומפיילר אחראי על פעולת ה garbage collector.
ד. קוד שקומפל על מערכת הפעלה אחת לא יכול לרוץ על מערכת הפעלה אחרת.
ה. מלבד תשובה זו יש יותר מתשובה נכונה אחת.
ו. מלבד תשובה זו, כל התשובות לא נכונות.

15. שאלה 15:

```
public class Test {
	public static int i = 0;
	public static void main(String[] args)
	{
		Display display = Display.getDefault();
		Shell shell = new Shell(display);
		shell.setLayout(new FillLayout(SWT.VERTICAL));
		Button b1 = new Button(shell, SWT.PUSH);
		b1.setText(""+i);
		Button b2 = new Button(shell, SWT.PUSH);
		b2.setText(""+i);

		b1.addSelectionListener(new ButtonHandler());
		b2.addSelectionListener(new ButtonHandler());
		shell.pack();
		shell.open();
		while (!shell.isDisposed()) {
		if (!display.readAndDispatch()) display.sleep();}
		display.dispose();
	}

	public static class ButtonHandler implements SelectionListener {
		public static int j = 1;
		@Override
		public void widgetDefaultSelected(SelectionEvent e) {}
		@Override
		public void widgetSelected(SelectionEvent e) {
			if (e.getSource() instanceof Button) {
				Button b = (Button) e.getSource();
				i++; // **
				j++; // ***
				b.setText("" + i * j);
			}}}
}
```

במקרה בו הקוד מתקמפל ורץ, הניחו כי המשתמשת לוחצת על הכפתור הראשון ולאחר מכן על הכפתור השני. בחר/י בתשובה הטובה ביותר:

א. יש שגיאת קומפילציה יחידה בשורה **.
ב. יש שגיאת קומפילציה יחידה בשורה ***.
ג. יש שגיאת קומפילציה גם בשורה ** וגם בשורה ***.
ד. הקוד מתקמפל ורץ, ולאחר לחיצות המשתמשת מופיע על שני הכפתורים המספר 6
ה. הקוד מתקמפל ורץ, ולאחר לחיצות המשתמשת מופיע 2 על הכפתור הראשון ו 6 על השני.
ו. הקוד מתקמפל ורץ, ולאחר לחיצות המשתמשת מופיע על שני הכפתורים המספר 2.

16. שאלה 16

```
public class A {
	public static String st = "A";
	public int i = 1;

	public void foo() {
		System.out.print(st + " " + i + " ");
	}
}

public class B extends A {
	public static String st = "B";
	public int i = 2;
}

public class C {
	public static void main(String[] args) {
		A a = new B();
		B b = new B();
		a.foo();
		b.foo();
	}
}
```

בחר/י בתשובה הטובה ביותר:

א. התכנית מדפיסה A 1 B 2
ב. התוכנית מדפיסה A 1 B 1
ג. התכנית מדפיסה B 2 B 2
ד. התכנית מדפיסה B 1 B 1
ה. התכנית מדפיסה A 1 A 1
ו. התכנית מדפיסה B 1 B 2

17. שאלה 17

מבין הטענות הבאות, בחר/י בתשובה הטובה ביותר:
טענה 1: פעולת casting בקוד יכולה לגרום לשגיאת קומפילציה.
טענה 2: פעולת casting בקוד יכולה לגרום לשגיאת זמן ריצה.
טענה 3: הפעולה (B)a תצליח (תתקמפל ותרוץ) רק אם a הוא מטיפוס סטטי B ומצביע לאובייקט מטיפוס דינאמי B.

א. רק טענה 1 נכונה.
ב. רק טענה 2 נכונה
ג. רק טענה 3 נכונה.
ד. רק טענות 2+1 נכונות
ה. רק טענות 3+1 נכונות.
ו. רק טענות 3+2 נכונות.
ז. כל הטענות נכונות.
ח. כל הטענות לא נכונות.

18. שאלה 18

```
public class A {
     public static void main(String[] args){
          Integer[] ints = new Integer[]{1, 2, null, 4};
          try{
               for (Integer i : ints){
                    try{
                         if (i %2 == 0){
                              System.out.print(i + "_");
                         }
                    }
                    finally{
                         System.out.print("#_");
                    }
               }
          }
          catch (NullPointerException exp){
               System.out.print("@_");
          }
          finally{
               System.out.print("%_");
          }
     }
}
```

מה יקרה בעת הרצת התוכנית? בחר/י בתשובה הטובה ביותר

א. יודפס _%_@_#_#_2_# והתוכנית תסתיים ללא שגיאות.
ב. יודפס _%_#_#_2_# והתוכנית תסתיים ללא שגיאות.
ג. יודפס @_#_2 והתוכנית תסתיים ללא שגיאות.
ד. יודפס _%_@_#_2 והתוכנית תסתיים ללא שגיאות.
ה. יודפס _%_4_#_#_2_# והתוכנית תסתיים ללא שגיאות.
ו. יודפס _4_#_2 והתוכנית תסתיים ללא שגיאות.
ז. בזמן ריצת התוכנית תיזרק שגיאת זמן ריצה מסוג NullPointerException.

19. שאלה 19:
מה יודפס בהרצת הקוד הבא? בחר/י בתשובה הטובה ביותר:

```
public class Point implements Comparable<Point>{
	int i;
	int j;

	public Point(int i, int j) {
		this.i = i;
		this.j = j;
	}

	@Override
	public int compareTo(Point other) {
		int cmpByI = Integer.compare(this.i, other.i);
		if (cmpByI == 0) {
			return Integer.compare(this.j, other.j);
		}
		return -cmpByI;	//there's a minus here!
	}

	@Override
	public int hashCode() {
		return this.i;
	}
	@Override
	public boolean equals(Object other) {
		return this.i == ((Point)other).i;
	}

	public String toString() {
		return "<" + this.i + "," + this.j + ">";
	}

	public static void main(String[] args) {
		//The elements in TreeSets are sorted in an ascending order
		Set<Point> points = new TreeSet<>();
		points.add(new Point(1,2));
		points.add(new Point(1,1));
		points.add(new Point(2,2));
		for (Point p : points) {
			System.out.print(p + " ");
		}
	}
}
```

א. <1,1> <1,2> <2,2>
ב. <1,2> <2,2> <1,1>
ג. <2,2> <1,1> <1,2>
ד. <2,2> <1,2> <1,1>
ה. <1,2> <1,1> <2,2>
ו. <1,1> <2,2> <1,2>
ז. <1,2> <2,2>
ח. <2,2> <1,2>

20. שאלה 20:

```
public class A {

	private int i = 1;
	public int j = 1;
	protected static int k = 1;
	private void foo() {System.out.println(i+j+k);}

	public static void main(String[] args)
	{
		A a = new A() {
			//private int i = 2; //*
			public int j = 2;
			protected void foo() {System.out.println(i+j+k);} //**
		};
		a.foo(); //***
	}
}
```

בחר/י בתשובה הטובה ביותר

א. יש שגיאת קומפילציה בשורה המסומנת ב ***.
ב. הקוד לא מתקמפל בגלל שגיאת קומפילציה בשורה **. אם נוציא מההערה את השורה המסומנת ב *, הקוד יתקמפל וידפיס 5.
ג. הקוד לא מתקמפל בגלל שגיאת קומפילציה בשורה **. אם נוציא מההערה את השורה המסומנת ב *, הקוד יתקמפל וידפיס 4.
ד. הקוד לא מתקמפל בגלל שגיאת קומפילציה בשורה **. אם נוציא מההערה את השורה המסומנת ב *, הקוד יתקמפל וידפיס 3.
ה. הקוד מתקמפל ומדפיס 4. אם נוציא מההערה את השורה המסומנת ב *, הקוד וידפיס 5.
ו. הקוד מתקמפל ומדפיס 4. אם נוציא מההערה את השורה המסומנת ב *, הקוד וידפיס 4.

21. שאלה 21:

הסטודנטית שי רוצה לכתוב פונקציה המקבלת שתי רשימות מטיפוס כלשהו, ומחזירה ערך בוליאני. אם שתי הרשימות מכילות בדיוק את אותם האיברים, הפונק' תחזיר `true`, ואחרת `false`. שתי הרשימות שהפונקציה מקבלת צריכות להיות מאותו הטיפוס הגנרי.
הקוד צריך להתנהג באופן באופן הבא:

```
List<Integer> lst1 = new ArrayList<>();
List<Integer> lst2 = new ArrayList<>();
List<String> lst3 = new ArrayList<>();
boolean res1 = func(lst1, lst2); // this compiles
boolean res2 = func(lst1, lst2);  // this doesn't compile
boolean res3 = func(lst1, lst3); // this doesn't compile
```

שי חשבה על מספר אופציות לחתימת הפונקציה:

אופציה 1: `public static <T> boolean func(List<T> list1, List<T> list2)`
אופציה 2: `public static boolean func(List<?> list1, List<?> list2)`
אופציה 3: `public static boolean func(List<Object> list1, List<Object> list2)`

בחר/י בתשובה הטובה ביותר:

א. רק אופציה 1 מתאימה.
ב. רק אופציה 2 מתאימה.
ג. רק אופציה 3 מתאימה.
ד. רק אופציות 1+2 מתאימות.
ה. רק אופציות 1+3 מתאימות
ו. רק אופציות 2+3 מתאימות.
ז. כל אופציות מתאימות.
ח. כל אופציות לא מתאימות.