

## בחינה בתוכנה 1

סמסטר א' תש"ף, מועד ב', 3 במרץ 2020  
לנה דנקין, שי גרשטיין, איתי יצחק

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 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 מסוימים. 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 על גבי טופס הבחינה במקומות המסומנים. 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 – התשובה נקבעת אך ורק על פי התשובה המסומנת בטופס התשובות.

יש לצרף את טופס המבחן למחברת הבחינה. מחברת ללא טופס עזר תפסל. סימוני תשובות 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 מחשבוני 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 מכונית ובין אלקטרונית או בכל דרך אחרת כל חלק שהוא מטופס הבחינה.

בהצלחה!

בשאלות 1-7 נממש את המחלקה AccessOrderMap אשר מהווה מבנה נתונים גנרי מסוג Map השומר על סדר הגישות למפתחות שלו. נרצה לדעת בכל רגע נתון מהו סדר הגישה למפתחות השונים שנשמרים ב Map, כאשר גישה מוגדרת להיות כל גישה שהיא למפתח – הכנסה, שליפה, בדיקת קיום וכו'. לצורך השאלה, הניחו כי כל השירותים שנורשו מ HashMap ואין אליהם התייחסות בשאלות 1-7 נדרסו ב AccessOrderMap ע"י מימוש אשר זורק חריג שהוא unchecked. הניחו כי לא נעשה בהם שום שימוש בשאלות אלה, ועליכם להתעלם מקיומם בעת המענה על השאלות השונות. עליכם לקרוא את כל השאלות על מנת להבין את הדרישות מהמחלקה. במענה על כל סעיף, הניחו כי הסעיפים הקודמים ממומשים נכון על פי החוזים, דוגמאות הריצה וההגדרות המילוליות בתרגיל. לפניכם דוגמת שימוש במחלקה. שימו לב שלולאת ה for בסוף הקוד נמצאת בהערה, ועליכם להוציא אותה מההערה בשאלות 6+7.

```
public class Tester {
    public static void main(String[] args) {
        AccessOrderMap<String, Integer> mMap = new AccessOrderMap<>();
        mMap.put("a", 1);
        mMap.put("b", 2);
        mMap.put("c", 3);
        System.out.println("(1) : " + mMap.getMostRecentlyAccessedKey());
        mMap.get("a");
        System.out.println("(2) : " + mMap.getMostRecentlyAccessedKey());
        mMap.get("d");
        System.out.println("(3) : " + mMap.getMostRecentlyAccessedKey());
        mMap.put("d", 4);
        mMap.put("a", 1);
        System.out.println("(4) : " + mMap.size());
        System.out.println("(5) : " + mMap.getMostRecentlyAccessedKey());
        mMap.remove("d");
        System.out.println("(6) : " + mMap.getMostRecentlyAccessedKey());
        mMap.remove("a");
        System.out.println("(7) : " + mMap.getMostRecentlyAccessedKey());
        mMap.remove("k");
        System.out.println("(8) : " + mMap.getMostRecentlyAccessedKey());
        System.out.println("(9) : " + mMap.size());

        /*
        for (String s : mMap) { // Uncomment this code for Q6+Q7
            System.out.print(s + " ");
        }
        */
    }
}
```

עבור ריצת תוכנית זו, זה הפלט הנדרש:

- (1) : c
- (2) : a
- (3) : a
- (4) : 4
- (5) : a
- (6) : a
- (7) : c
- (8) : c
- (9) : 2

זהו שלד המחלקה אותה נממש:

```
/**
 * @impl_inv: size() == accessList.size();
 */
public class AccessOrderMap /***/ Q1 ****/ {
    private /***/ Q1 ****/ accessList;

    public AccessOrderMap() {
        this.accessList = new ArrayList<>();
    }

    private void updateAccessForKey(T key) {
        //reminder: if element is not in the list, remove does nothing
        accessList.remove(key);
        accessList.add(0, key);
    }

    public T getMostRecentlyAccessedKey() {
        return accessList.get(0);
    }

    @Override
    /* @post: getMostRecentlyAccessedKey() == key */
    public K put(T key, K value) { //Q2
        // your implementation here
    }

    @Override
    /* @post: size() == prev(size()) */
    public K get(Object key) { //Q3
        // your implementation here
    }

    @Override
    public int size() { //Q4
        // your implementation here
    }

    public K remove(Object key) { { //Q5
        // your implementation here
    }
}
```

1. שאלה 1:

מבין האופציות הבאות השלימו את הגדרת המחלקה AccessOrderMap ואת הגדרת השדה שלה.

```
public class AccessOrderMap extends HashMap<T, K>{ //op1
    private List<T> accessList;

public class AccessOrderMap extends HashMap<T, K>{ //op2
    private List accessList;

public class AccessOrderMap<T,K> extends HashMap<T,Integer><>{ //op3
    private List<T> accessList;

public class AccessOrderMap<T,K> extends HashMap<T,Integer>{ //op4
    private List accessList;

public class AccessOrderMap<T,K> extends HashMap<T,K>{ //op5
    private List<T> accessList;

public class AccessOrderMap<T,K> extends HashMap<T,K>{ //op6
    private List accessList;
```

בחר/י בתשובה הטובה ביותר:

נימוק:

- א. רק אופציה 1 מתאימה.
- ב. רק אופציה 2 מתאימה.
- ג. רק אופציה 3 מתאימה.
- ד. רק אופציה 4 מתאימה.
- ה. **רק אופציה 5 מתאימה.**
- ו. רק אופציה 6 מתאימה.
- ז. יש יותר מאופציה מתאימה אחת.
- ח. כל האופציות לא מתאימות.

2. שאלה 2:

לפניכם מימוש אפשרי עבור השירות put.

```
public K put(T key, K value) {  
    updateAccessForKey(key);  
    return super.put(key, value);  
}
```

טענה 1: מימוש זה מקיים את משתמר המחלקה.  
טענה 2: אם נחליף את השורה הראשונה ב `accessList.add(0,key)`, מימוש זה ישמור על משתמר המחלקה.  
טענה 3: ניתן להוסיף לשירות את התנאי `@pre: this.contains(key) == False`. אם נוסיף תנאי זה, גם טענה 1 וגם טענה 2 יהיו נכונות, ולא תהיה פגיעה באופן השימוש בשירות.  
בחר/י בתשובה הטובה ביותר:

נימוק:

- 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 לא נכונות.

3. שאלה 3:

לפניכם מימוש אפשרי עבור השירות get. תזכורת: השירות get של HashMap מחזיר null עבור מפתח שאינו קיים. כמו כן, אין שום בעיה להכניס null כמפתח ב HashMap.

```
public K get(Object key) {  
    updateAccessForKey((T)key);  
    return super.get(key);  
}
```

טענה 1: מימוש זה כמו שהוא מפר את המשתמר (invariant) של המחלקה.  
טענה 2: המימוש תקין. אם נוריד את השורה הראשונה של המימוש זה יפר את המשתמש (invariant) של המחלקה.  
טענה 3: תיתכן שגיאת זמן ריצה בהרצת השירות get.  
בחר/י בתשובה הטובה ביותר:

נימוק:

- 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.

4. שאלה 4:

```
public int size() { return super.size(); } //op1
public int size() { return this.size(); } //op2
public int size() { return accessList.size(); } //op3
```

אילו מבין המימושים מתאימים לדוגמת הריצה בתוכנית Tester וגם שומרים על חוזה המחלקה?  
בחר/י בתשובה הטובה ביותר:

נימוק:

- א. רק מימוש 1
- ב. רק מימוש 2
- ג. רק מימוש 3
- ד. רק מימושים 1+2
- ה. רק מימושים 1+3
- ו. רק מימושים 2+3
- ז. כל המימושים מתאימים.
- ח. כל המימושים לא מתאימים.

5. שאלה 5:

השלימו את המימוש של remove מבין שורות קוד הבאות:

```
1. accessList.remove(key);
2. if (accessList.contains(key)) {
3. updateAccessForKey((T)key);
4. return this.remove(key);
5. return super.remove(key);
6. }
```

בחר/י במימוש הנכון:

נימוק:

- א. 1,4
- ב. 1,5
- ג. 2,3,4,6
- ד. 2,3,5,6
- ה. 3,4
- ו. 3,5
- ז. מלבד תשובה זו יש לפחות שתי תשובות נכונות.
- ח. מלבד תשובה זו, כל התשובות לא נכונות.

6. שאלה 6:

כעת, נוציא את לולאת ה for אשר מופיעה ב Tester מהערה. אילו שינויים יש לבצע ב AccessOrderMap על מנת שלולאת ה for תתקמפל ותרוץ?  
טענה 1: עלינו להוסיף implements Iterator<T> להצהרה על המחלקה, ולממש את שירותי הממשק.  
טענה 2: עלינו להוסיף implements Iterable<T> להצהרה על המחלקה, ולממש את שירותי הממשק.  
טענה 3: ניתן להתאים את הקוד ללא הוספת מימושי ממשקים למחלקה כך שלולאת ה for תתקמפל ותרוץ. אילו מבין הטענות נכונה?  
בחר/י בתשובה הטובה ביותר:

נימוק:

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

7. שאלה 7:

נרצה להשלים את המימוש של AccessOrderMap כך שהלולאת for תחזיר את כל האיברים על פי סדר הגישה אליהם – מהמאוחר ביותר למוקדם ביותר. בפרט, נרצה שב Tester לולאת ה for תייצר את הפלט הבא:

c b

טענה 1: על מנת לאפשר את ריצת הלולאה, ניתן להסתפק בהוספת פונק' אחת שמכילה שורת קוד אחת ב Java (כלומר, שורת קוד אחת שמסתיימת ב ;) למחלקה AccessOrderMap ללא קוד נוסף.  
טענה 2: על מנת לאפשר את ריצת הלולאה, אנחנו חייבים לעשות שימוש ב super.keys().  
טענה 3: לא ניתן לכתוב מימוש לאיטרטור של AccessOrderMap במחלקה נפרדת.

נימוק:

- 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.

8. שאלה 8:

נתון הקוד של המחלקה Box ובה שירות main בו ההגדרה של המשתנה b אינה נתונה במלואה.

```
public class Box<T,V>{  
    public <K> void func1(List<K> lst1, List<K> lst2) {};  
    public void func2(List<T> lst1, List<V> lst2) {};  
    public void func3(List<?> lst1, List lst2) {};  
  
    public static void main(String[] args) {  
        Box<*****> b = new Box<>();  
        List<String> strList = new ArrayList<>();  
        List<Integer> intList = new ArrayList<>();  
        b.func1(strList, intList); /**  
        b.func2(strList, intList); /**  
        b.func3(strList, intList); /**  
    }  
}
```

טענה 1: ניתן להגדיר את b כך שהקריאה ל func1 (מסומנת ב \*) תתקמפל.  
טענה 2: ניתן להגדיר את b כך שהקריאה ל func2 (מסומנת ב \*\*) תתקמפל.  
טענה 3: ניתן להגדיר את b כך שהקריאה ל func3 (מסומנת ב \*\*\*) תתקמפל.

נימוק:

- 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.



9. שאלה 9:

לפניכם הקוד של התוכנית A ומספר טענות.

```
public class A{
    public static A singleA = new A();

    public static void recFunc(A a) {
        if (a != null) { recFunc(a.next); }
    }

    public static void iterFunc(A a) {
        for (A tmp = singleA; tmp != null; tmp = tmp.next) { }
    }

    private A next = null;
    @Override
    public void finalize() {
        recFunc(singleA);
        iterFunc(singleA);
        this.next = singleA;
        singleA = this;
    }

    public static void main(String[] args) {
        while (true) { A a = new A(); }
    }
}
```

1: אם ה heap גדול כרצונינו וה stack מוגבל בגודלו, התוכנית תזרוק StackOverflowException.  
2: אם ה stack גדול כרצונינו וה heap מוגבל בגודלו, התוכנית תזרוק OutOfMemoryException.  
3: בפעם הראשונה ש finalize נקראת קיימים רק 2 אובייקטים מטיפוס A בזכרון.  
בחר/י בתשובה הטובה ביותר:

נימוק:

- 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:

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

    @Override
    public B clone() throws
        CloneNotSupportedException {
        return (B)super.clone();
    }

    public static void main(String[] args) throws
        CloneNotSupportedException {
        B b = new B(1);
        B cloneB = (B)b.clone();
    }
}
```

לפניכם מספר טענות:

- טענה 1: הקוד הנוכחי יזרוק CloneNotSupportedException בהרצתו. בשביל למנוע את זה, צריך להוסיף implements Cloneable להגדרת המחלקה B.
- טענה 2: ניתן לממש את clone לא שימוש super.clone וללא מימוש הממשק Cloneable.
- טענה 3: הקוד כפי שהוא רץ ללא שגיאות.  
בחר/י בתשובה הטובה ביותר:

נימוק:

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

11. שאלה 11:

```
public class A {  
    public static void check(Object o, String s){}  
}  
public class B extends A{  
    /* public static void check(Object o, String s){} \\ option 1*/  
    /* public static void check(Object o, Object s){} \\ option 2*/  
    /* public static void check(String o, Object s){} \\ option 3*/  
    /* public static void check(String o, String s){} \\ option 4*/  
}
```

בחר/י בתשובה הטובה ביותר:

נימוק:

- א. אם נוציא את המתודה בשורה 1 option מהערה המחלקה B תתקמפל, שאר המתודות לא יתקמפלו.
- ב. אם נוציא את המתודה בשורה 2 option מהערה המחלקה B תתקמפל, שאר המתודות לא יתקמפלו.
- ג. אם נוציא את המתודה בשורה 3 option מהערה המחלקה B תתקמפל, שאר המתודות לא יתקמפלו.
- ד. אם נוציא את המתודה בשורה 4 option מהערה המחלקה B תתקמפל, שאר המתודות לא יתקמפלו.
- ה. ישנן 2 שורות בדיוק שאם נוציא אותן מהערה (כל שורה בנפרד) המחלקה B תתקמפל, שאר המתודות לא יתקמפלו.
- ו. ישנן 3 שורות בדיוק שאם נוציא אותן מהערה (כל שורה בנפרד) המחלקה B תתקמפל, המתודה הנוותרת לא תתקמפל.
- ז. אם נוציא מהערה את כל 4 השורות (כל שורה בנפרד) המחלקה B תתקמפל.

12. שאלה 12

מה יודפס בהרצת התוכנית הבאה?

```
public class StringPrints {  
    public static void main(String[] args) {  
        String s1 = "hello";  
        String s2 = s1 + " world";  
        String s3 = new String("hello");  
        String s4 = 'h'+ 'e'+ 'l'+ 'l'+ 'o'; // *  
  
        System.out.print(s1 == s2); // **  
        System.out.print(" ");  
        System.out.print(s1.equals(s3));  
        System.out.print(" ");  
        System.out.print(s4.equals(s1)); // ***  
    }  
}
```

בחר/י בתשובה הנכונה ביותר:

נימוק:

- א. false false false
- ב. false true false
- ג. false true true
- ד. true false false
- ה. true false true
- ו. true true true

- ז. הקוד אינו מתקמפל באחת מהשורות \*, \*\*, או \*\*\*.
- ח. הקוד מתקמפל אך זורק שגיאת זמן ריצה באחת מהשורות \*, \*\*, או \*\*\*.

13. שאלה 13

- טענה 1 - מחלקה אבסטרקטית יכולה לממש ממשק.
  - טענה 2 - ממשק יכול לרשת ממחלקה אבסטרקטית אם כל המתודות שלה אבסטרקטיות.
  - טענה 3 - מתודה default בממשק היא בהכרח סטטית.
  - טענה 4 - הטיפוס הדינמי של מחלקה אנונימית שממשת ממשק הוא הממשק עצמו.
- בחר/י את התשובה הנכונה ביותר:

נימוק:

- 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+2+3 נכונות.
- ו. רק טענות 1+3 נכונות.
- ז. רק טענות 2+3+4 נכונות.
- ח. כל הטענות נכונות.

14. שאלה 14

הסטודנטית ברית התבקשה לכתוב מחלקה אבסטרקטית כלשהי בשם A. ברית טענה כי ניתן להחליף את המחלקה האבסטרקטית בממשק. בחר/י את התשובה הנכונה ביותר:

נימוק:

- א. ברית צודקת רק אם ידוע שב A אין שדות מופע וגם אין בה מתודות מופע לא אבסטרקטיות בכלל.
- ב. ברית צודקת רק אם ידוע שב A אין שדות מופע וגם אין בה שדות סטטיים בכלל.
- ג. ברית טועה, ממשק לעולם לא יוכל להחליף מחלקה אבסטרקטית כי לא ניתן לבצע המרה כלפי מעלה (upcasting) לטיפוס של ממשק כפי שניתן לעשות לטיפוס של מחלקה אבסטרקטית.
- ד. ברית טועה אם ידוע שישנן מתודות מופע ציבוריות עם מימוש ב A.
- ה. **מלבד תשובה זו, כל התשובות לא נכונות.**
- ו. מלבד תשובה זו יש יותר מתשובה נכונה אחת.

15. שאלה 15

לפניכם 3 טענות הקשורות לממשקים:

- טענה 1: כל מתודת מופע בעלת מימוש במחלקה אבסטרקטית שירשת מ-Object שלא קוראת למתודה אחרת מאותה המחלקה ניתן להעביר עם אותו המימוש לממשק בתור מתודת default.
- טענה 2: ניתן להגדיר ממשק גנרי.
- טענה 3: ניתן להגדיר שדות מופע בממשקים, אך הם תמיד יהיו public ו-final

בחרו את התשובה הנכונה ביותר:

נימוק:

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

16. שאלה 16

שאלה זו מתייחסת למימוש של מחלקות A, B, C 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 = (C) b2; // line 2  
}
```

להלן מספר טענות לגבי קוד זה:

טענה 1: ניתן לממש את A, B ו-C כך ששתי השורות יתקמפלו ועלולה להיזרק שגיאת זמן ריצה בגלל casting רק בשורה 1.

טענה 2: ניתן לממש את A, B ו-C כך ששתי השורות יתקמפלו ועלולה להיזרק שגיאת זמן ריצה בגלל casting רק בשורה 2.

טענה 3: ניתן לממש את A, B ו-C כך ששתי השורות יתקמפלו ולא יכולה להיזרק שגיאת זמן ריצה בגלל casting.

נימוק:

הניחו שכאשר שורה מסוימת מורצת, השורה אחרת נמצאת בהערה.

בחרו את התשובה הנכונה ביותר:

א. אף טענה אינה נכונה

ב. רק טענה 1 נכונה

ג. רק טענה 2 נכונה

ד. רק טענה 3 נכונה

ה. רק טענות 1+2 נכונות

ו. רק טענות 1+3 נכונות

ז. רק טענות 2+3 נכונות

ח. כל הטענות נכונות

17. שאלה 17:

```
public class Outer {  
  
    private void f() { new NS().m(); }  
  
    private static void s() { new NI().n(); }  
  
    private static class NS {  
        private void m() {  
            f();  
            s();  
        }  
    }  
  
    private class NI {  
        private void n() {  
            f();  
            s();  
        }  
    }  
}
```

קבעו אילו מהמתודות n, m, s, f מתקמפלות. בבואכם לבדוק את הקומפילציה m ו n, הניחו שהמימוש של f ו s הוא תקין, כלומר, עליכם להתעלם מהמימוש בתוך f ו s ולבחון רק את אופן השימוש בחתימות שלהן.

נימוק:

- א. כל המתודות מתקמפלות.
- ב. כל המתודות לא מתקמפלות.
- ג. רק מתודה f לא מתקמפלת.
- ד. רק מתודה s לא מתקמפלת.
- ה. רק מתודה m לא מתקמפלת.
- ו. רק מתודה n לא מתקמפלת.
- ז. שתי מתודות לא מתקמפלות.
- ח. שלוש מתודות לא מתקמפלות.



18. שאלה 18:

מה יודפס לאחר הרצת הקוד שלפניכם?

```
public class A {
    int i;

    public A(int i) { this.i = i; }

    public class Engine{
        public A a;
        int i;

        public Engine(A a) {
            this.a = a;
            this.i = A.this.i;
        }

        public void f() {
            System.out.println(this.i);
            System.out.println(a.i);
            System.out.println(A.this.i);
        }
    }

    public static void main(String[] args) {
        A c = new A(1);
        Engine e = c.new Engine(new A(2));
        e.f();
    }
}
```

בחר/י בתשובה הטובה ביותר:

נימוק:

- א. 111
- ב. 121
- ג. 122
- ד. 112
- ה. 211
- ו. 212
- ז. 221
- ח. 222



19. שאלה 19:

```
public class A {  
    private static class B {  
        public static void f() { System.out.println("B"); }  
    }  
  
    private static class C extends B {  
        public static void f() { System.out.println("C"); }  
    }  
  
    public static void main(String[] args) {  
        C c = new C();  
        c.f();  
        B b2 = c;  
        b2.f();  
        c = (C) new B();  
        c.f();  
    }  
}
```

מה יקרה בהרצת הקוד הבא? בחר/י בתשובה הטובה ביותר

נימוק:

- א. ידפיס CBC
- ב. ידפיס CBB
- ג. ידפיס BBB
- ד. ידפיס CCC
- ה. ידפיס BC ויעוף על חריג
- ו. ידפיס BB ויעוף על חריג
- ז. ידפיס CB ויעוף על חריג
- ח. ידפיס CC ויעוף על חריג

20. שאלה 20:

```
public class A {
    String s = "A";

    public void f() { System.out.print(s); }

    private void g() {
        f();
        System.out.print(s);
    }

    public A() { g(); }
}

public class B extends A {
    String s = "B";

    public void f() { System.out.print(s); }

    public void g() {
        System.out.print(s);
        f();
    }

    public B() { f(); }

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

מה יקרה בהרצת התוכנית B? בחר/י בתשובה הטובה ביותר

נימוק:

- א. התוכנית לא מתקמפלת בכלל.
- ב. התוכנית עפה על חריג בשורה \*\*.
- ג. התוכנית מדפיסה AAB
- ד. התוכנית מדפיסה nullBA
- ה. התוכנית מדפיסה nullAB
- ו. התוכנית מדפיסה ABA
- ז. התוכנית מדפיסה nullBB
- ח. התוכנית מדפיסה ABB

21. שאלה 21:

```
public class MyGui {
    private static int j;

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        while (j < 1) {
            System.out.println("Enter a positive integer: ");
            j = s.nextInt();
        }
        s.close();
        Display display = Display.getDefault();
        Shell shell = new Shell(display);
        shell.setLayout(new RowLayout(SWT.VERTICAL));
        Button ok = new Button(shell, SWT.PUSH);
        ok.setText(j);
        ok.addSelectionListener(*****); //missing code
        shell.pack();
        shell.open();
        while (shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    }
}
```

אנו מניחים שהתוכנית מקבלת ארגומנט אחד שהוא מחרוזת המייצגת מספר חיובי שלם. כמו כן, אנו מניחים שהמשתמש מזין מספר שלם חיובי שיוצב ב-j. נרצה כי בכל לחיצה על הכפתור הערך של השדה j יגדל ב-args[0] (ניתן להניח שמערך הארגומנטים לא ישתנה לאורך ריצת התוכנית), ובנוסף נרצה שערכו החדש של j יופיע על הכפתור. טענה 1: ניתן לממש התנהגות זו על ידי כתיבת מאזין (Listener) שמוגדר כמחלקה אנונימית (בקוד החסר המסומן בכוכביות) אשר מממשת את המנשק SelectionListener. טענה 2: ניתן לממש התנהגות זו על ידי כתיבת מאזין שמוגדר כמחלקה מקוונת סטטית (המחלקה מקוונת בתוך המחלקה MyGui, אך לא בתוך המתודה main) אשר מממשת את המנשק SelectionListener, וליצור מופע שלו בקוד החסר המסומן בכוכביות. טענה 3: ניתן לממש התנהגות זאת על ידי כתיבת מאזין שמממש את המנשק SelectionListener ומוגדר כמחלקה חדשה בקובץ נפרד, וליצור מופע שלו בקוד החסר המסומן בכוכביות. בכל הטענות, הניחו כי אין לשנות חלקים נוספים בקוד שלא הוזכרו בטענה. בחר/י בתשובה הטובה ביותר:

נימוק:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה, והיא תמשיך להיות נכונה גם אם j יוגדר כלא סטטי.
- ג. רק טענות 2 ו-3 נכונות.
- ד. כל הטענות נכונות.
- ה. רק טענות 1 ו-2 נכונות, אך יפסיקו להיות נכונות אם j יוגדר להיות לא סטטי.
- ו. כל הטענות לא נכונות, וימשיכו להיות לא נכונות גם אם j יוגדר להיות לא סטטי.
- ז. רק טענה 2 נכונה, אך אם j יוגדר כלא סטטי קבוצת הטענות הנכונות תשתנה.
- ח. כל הטענות לא נכונות, אך אם j יוגדר להיות לא סטטי, לפחות טענה אחת תהפוך להיות נכונה.