GuyAI
מבית GOOL בגרויות

סוג הבחינה: בחינת גמר ממלכתית
מועד הבחינה: קיץ 2026 מועד א'
סמל השאלון: 97105
שם התלמיד/ה: _____

מבנה נתונים ותכנות מונחה עצמים

הנדסת תוכנה · הנדסאים וטכנאים

הוראות

א. משך הבחינה: ארבע שעות וחצי.

ב. מבנה השאלון ומפתח ההערכה: **בשאלון זה שני מבחנים, עליכם לענות על מבחן אחד בלבד בהתאם למוסד הלימודים:** מבחן ב- C# (עמוד 2) מבחן ב- JAVA (עמוד 12) בכל מבחן תשע שאלות. **חלק א' – 32 נקודות** שאלות 1–3: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 16 נקודות. **חלק ב' – 32 נקודות** שאלות 4–6: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 16 נקודות. **חלק ג' – 36 נקודות** שאלות 7–9: יש לענות על **שתי** שאלות בלבד. ערך כל השאלה 18 נקודות. **בסך הכול: 100 נקודות.**

ג. חומר עזר מותר בשימוש: 1. מחשבון (אין להשתמש במחשב כף יד או במחשבון עם תקשורת חיצונית). 2. קלסר **אחד בלבד** עם **חומר ההרצאות**. אין להוציא דפים מהקלסר. **אין לצרף ספרים או חוברות עם פתרונות.**

ד. הוראות מיוחדות: 1. יש לקרוא בעיון את ההנחיות בדף השער ואת כל שאלות המבחן, ולוודא שהן מובנות. 2. את התשובות יש לכתוב בצורה מסודרת, בכתב יד ברור ונקי (גם בכך תלויה הערכת המבחן). 3. יש להשאיר את העמוד הראשון במחברת הבחינה ריק. בסיום המבחן יש לרשום בעמוד זה את מספרי התשובות לבדיקה. התשובות ייבדקו לפי סדר כתיבתן בעמוד זה. לא ייבדקו תשובות עודפות. 4. יש לכתוב את התשובות במחברת הבחינה **בעט בלבד**, בכתב יד ברור. 5. יש להתחיל כל תשובה בעמוד חדש ולציין את מספר השאלה ואת הסעיף. אין צורך להעתיק את השאלה עצמה. 6. טיוטה יש לכתוב במחברת הבחינה בלבד. יש לרשום את המילה "טיוטה" בראש העמוד ולהעביר עליו קו כדי שלא ייבדק. 7. יש להציג פתרון מלא ומנומק, כולל חישובים לפי הצורך. הצגת תשובה סופית ללא שלבי הפתרון לא תזכה בניקוד. 8. יש להסביר בפירוט כל תוכנית שנכתבה, **תוכנית ללא הסבר מפורט לא תזכה בניקוד.** 9. אם לדעתכם חסר בשאלה נתון, יש לציין זאת ולהוסיף נתון מתאים שיאפשר לכם להמשיך בפתרון השאלה, נמקו את בחירתכם. **חל איסור מוחלט להוציא שאלון או מחברת בחינה מחדר הבחינה!** בנספחי הבחינה מובאים ממשקים (interface) הכוללים פונקציות ומחלקות עזר, יש להקפיד ולהשתמש בהן בשאלות הרלוונטיות בבחינה על-מנת להימנע מאיבוד ניקוד. לשאלון מצורפים נספח ממשקים לבחינה JAVA, נספח ממשקים לבחינה C# ומילון עזר.

השאלות בשאלון זה מנוסחות בלשון רבים, אף על פי כן על כל תלמידה וכל תלמיד להשיב עליהן באופן אישי.

בהצלחה!

השאלות

1. "תור צומח בקיצוניות" הוא תור שבו כל איבר גדול מסכום כל האיברים שקדמו לו בתור.

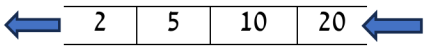
לדוגמה: התור שלפניכם הוא "צומח בקיצוניות"

כי:

- 2 < 5

- 5+ 2 < 10

- 2+5+ 10 < 20



א. כתבו פעולה המקבלת תור של מספרים שלמים חיוביים q ומספר שלם num. הפעולה תבדוק אם המספר num גדול מסכום כל האיברים הנמצאים בתור q. אם כן, הפעולה תחזיר ערך true, ואם לא - הפעולה תחזיר ערך false.

כותרת הפעולה:

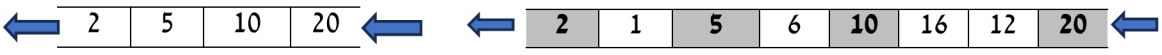
Java:

```
public static boolean isGreater(Queue<Integer> q, int num)
```

C#:

```
public static bool IsGreater(Queue<int> q, int num)
```

(6 נקודות)



ב. כתבו פעולה המקבלת תור של מספרים שלמים חיוביים q. הפעולה תחזיר תור חדש המבוסס על איברי q (בסדר הופעתם) המקיים את הגדרת "תור צומח בקיצוניות". יש לשמור על מצב התור q כפי שהתקבל.

לדוגמה, עבור התור q: 2, 1, 5, 6, 10, 16, 12, 20

הפעולה תחזיר תור חדש: 2, 5, 10, 20

כותרת הפעולה:

Java:

```
public static Queue<Integer> buildExtreme(Queue<Integer> q)
```

C#:

```
public static Queue<int> BuildExtreme(Queue<int> q)
```

יש להשתמש בפעולה שכתבתם בסעיף א'.

(8 נקודות)

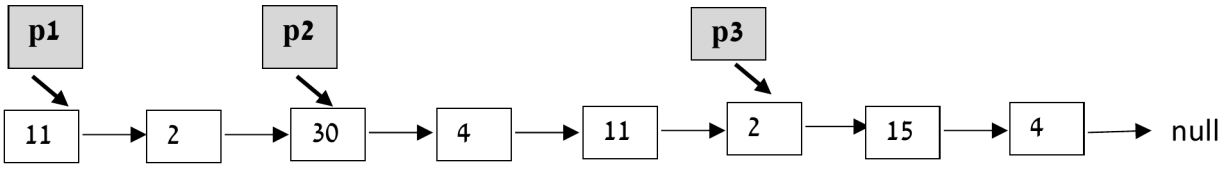
ג. מהי הסיבוכיות של הפעולות שכתבתם בסעיפים א' ו-ב'? **הסבירו את תשובתכם.**

(2 נקודות)

2. הגדרה: נתונה שרשרת של מספרים שלמים. "**עוצמה**" (Magnitude) של חוליה p היא מספר החוליות העוקבות לה ברצף, שערכן **קטן** מערך החוליה p.

לדוגמה:

- ה"עוצמה" של החוליה p1 היא 1;
- ה"עוצמה" של החוליה p2 היא 5;
- ה"עוצמה" של החוליה p3 היא 0.



נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

א. כתבו פעולה המקבלת הפניה לחוליה p בשרשרת חוליות של מספרים שלמים. הפעולה תחזיר את ה"עוצמה" של החוליה p.

כותרת הפעולה:

Java:

```
public static int getMagnitude(Node<Integer> p)
```

C#:

```
public static int GetMagnitude (Node<int> p)
```

(7 נקודות)

ב. נתונה שרשרת חוליות באורך n. מהו הערך המקסימלי האפשרי ל"עוצמה" של חוליה כלשהי בשרשרת זו? **הסבירו את תשובתכם.**

(2 נקודות)

ג. כתבו פעולה חיצונית המקבלת הפניה לחוליה הראשונה של שרשרת חוליות chain. הפעולה תדפיס את ערכי החוליות לפי סדר ה"עוצמה" שלהן: תחילה את אלו שעוצמתן 0, לאחר מכן את אלו שעוצמתן 1 וכן הלאה. כותרת הפעולה:

Java:

```
public static void printByMagnitude(Node<Integer> chain)
```

C#:

```
public static void PrintByMagnitude(Node<int> chain)
```

(7 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

3. נתונה היררכיית מחלקות וממשקים:

- **הממשק** IFlier כולל את כותרת הפעולה
`public void fly()` (ב-C#: `public void Fly()`).
- **הממשק** IGPS כולל את כותרת הפעולה
`public void navigate(double x, double y)` (ב-C#: `public void Navigate(double x, double y)`).
- **המחלקה** Device היא מחלקת בסיס (על), ולה תכונה
`protected String serialNum` (ב-C#: `protected string serialNum`).
- **המחלקה** Drone יורשת מ-Device וממשת את הממשקים IFlier ו-IGPS.
המחלקה גם כוללת פעולה ייחודית
`public void takePhoto()` (ב-C#: `public void TakePhoto()`).
- **המחלקה** SmartPhone יורשת מ-Device וממשת את הממשק IGPS.
המחלקה גם כוללת פעולה ייחודית
`public void call()` (ב-C#: `public void Call()`).
- **המחלקה** PaperPlane ממשת רק את הממשק IFlier.

הערה: בכל מחלקה מוגדרת פעולה בונה ללא פרמטרים המאתחלת את התכונות.

א. ציירו עץ ירושה המציג את הקשרים בין כל המחלקות והממשקים שצוינו לעיל.

הנחיות לציור:

- השתמשו בציון ברור עבור מחלקות וממשקים (ציינו מי הוא interface ומי class).
- השתמשו בחץ רציף לסימון **ירושה** (---->).
- השתמשו בחץ מקווקו לסימון **מימוש ממשק** (- - >).

(3 נקודות)

ב. עבור המחלקות Device, Drone, SmartPhone ו-PaperPlane, כתבו את כותרת המחלקה ואת כותרות הפעולות (__אין צורך לממש את הפעולות__).

(4 נקודות)

ג. עבור כל אחת מההוראות שלפניכם, קבעו אם היא **תקינה** או **אינה תקינה**. אם ההוראה אינה תקינה, הסבירו מדוע וציינו אם זו **שגיאת הידור** (קומפילציה) או **שגיאת זמן ריצה**.

Java:

```
6. Device d = new Drone();

7. IFlier f = new IFlier();

8. IGPS g = new SmartPhone();

9. IFlier f2 = new Drone();
   ((Drone)f2).fly();

10.    Device d2 = new Device();
   ((Drone)d2).fly();
```

C#:

```
1. Device d = new Drone();

2. IFlier f = new IFlier();

3. IGPS g = new SmartPhone();

4. IFlier f2 = new Drone();
   ((Drone)f2).Fly();

5. Device d2 = new Device();
   ((Drone)d2).Fly();
```

(5 נקודות)

ד. כתבו פעולה חיצונית המקבלת כפרמטר מערך items של עצמים מטיפוס Object. הפעולה תעבור על המערך ותדפיס "Fly" עבור כל עצם שמממש את הממשק IFlier, ותדפיס "Navigate" עבור כל עצם שמממש את הממשק IGPS.

(4 נקודות)

4. בשאלה זו עליכם לכתוב פעולות חיצוניות המטפלות בשרשרת חוליות של מספרים שלמים.

אין להשתמש במבני נתונים נוספים (כגון מערך, תור וכו') ואין ליצור חוליות חדשות.

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

א. כתבו פעולה חיצונית המקבלת הפניה לחוליה הראשונה בשרשרת ch. הפעולה תחזיר **הפניה** (מצביע) לחוליה **האחרונה** בשרשרת שערכה הוא **זוגי**. אם אין בשרשרת שום חוליה שערכה זוגי, הפעולה תחזיר null. כותרת הפעולה:

Java:

```
public static Node<Integer> lastEvenNode(Node<Integer> ch)
```

C#:

```
public static Node<int> LastEvenNode(Node<int> ch)
```

(4 נקודות)

ב. כתבו פעולה חיצונית המקבלת הפניה לחוליה הראשונה בשרשרת ch. הפעולה תאתר את החוליה הזוגית **האחרונה** בשרשרת ותעביר אותה (__על ידי שינוי מצביעים בלבד!__) כך שתהיה החוליה הראשונה בשרשרת. הפעולה תחזיר הפניה לחוליה הראשונה החדשה. כותרת הפעולה:

Java:

```
public static Node<Integer> lastEvenToFront(Node<Integer> ch)
```

C#:

```
public static Node<int> LastEvenToFront(Node<int> ch)
```

יש להשתמש בפעולה lastEvenNode מסעיף א'.

(6 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

ג. נתונה הפעולה `countEven(Node<Integer> ch)` (ב-`CountEven(Node<int> ch)` :C#`)
המחזירה את מספר הערכים הזוגיים בשרשרת (__אין צורך לממש את הפעולה__).
כתבו פעולה חיצונית המקבלת הפניה לחוליה הראשונה בשרשרת ch. הפעולה תשנה את השרשרת כך ש**כל הערכים הזוגיים** יופיעו בתחילתה (הסדר הפנימי שלהם אינו משנה).
הפעולה תחזיר הפניה לחוליה הראשונה החדשה. כותרת הפעולה:

Java:

```
public static Node<Integer> allEventsToFront(Node<Integer> ch)
```

C#:

```
public static Node<int> AllEventsToFront(Node<int> ch)
```

יש להשתמש בפעולה countEven ובפעולה lastEvenToFront.

(6 נקודות)

5. במרכז לוגיסטי גדול מנהלים את נתוני כוח האדם. לכל עובד במרכז יש **שם** ו**מספר עובד**. **מספר העובד הוא ייחודי וניתן באופן אוטומטי** (העובד הראשון במערכת מקבל את המספר 1, השני 2 וכן הלאה).

להלן הגדרות התפקידים במרכז:

תפקיד	תיאור תפקיד
סדרן (Dispatcher)	עובד האחראי על שיבוץ משלוחים לאזורי הפצה. נוסף לתכונות העובד הכלליות, לסדרן יש תכונה **מספר עמדה** (stationNum) מטיפוס שלם.
נהג (Driver)	עובד בעל תכונה **סוג רישיון** (licenseType) מטיפוס מחרוזת. לדוגמה: "C" , "E" , "C1".
נהג משאית (TruckDriver)	נוסף לתכונות הנהג, יש לו תכונה **משקל מקסימלי מורשה** (maxWeight) מטיפוס ממשי, המציינת את המשקל המקסימלי שהוא רשאי לשנע.
מנהל משמרת (Manager)	מנהל המשמרת הוא עובד לכל דבר. הוא האחראי על צוות עובדים. למנהל יש **מערך** בשם team המכיל עד 20 העובדים הכפופים לו (מכל הסוגים) ותכונה current המייצגת את מספרם בפועל.

א. שרטטו תרשים UML המתאר את הקשרים בין כל המחלקות. על התרשים לשקף את __עקרונות התכנות מונחה העצמים__ - הגדירו מחלקת על בשם **Employee** ובצעו הורשה והכלה בהתאם לתיאור.

השתמשו בחץ רציף ◄--- לירושה וחץ עם מעוין ◆--- להכלה.

(4 נקודות)

ב. לכל אחת מהמחלקות כתבו:

1. כותרת מחלקה;
2. תכונות - הקפידו על הרשאות גישה (private/protected/public) לפי עקרונות של תמ"ע ועל שימוש ב-static היכן שנדרש;
3. פעולה בונה (Constructor) המקבלת את הפרמטרים הרלוונטיים לאתחול העצם.

שימו לב: מספר העובד נוצר אוטומטית!

(6 נקודות)

אפשר להניח שקיימות פעולות get/set בכל אחד מהמחלקות - אין צורך לממש אותן. (ב-C#: Get/Set)

ג. המחלקה LogisticsCenter כוללת מערך של כל עובדי המרכז בשם allEmployees ומונה total המציין מספר העובדים בפועל.

הוסיפו למחלקה את הפעולות האלה:

1. פעולה בשם heavyDutyCheck (ב-C#: HeavyDutyCheck). הפעולה תקבל כפרמטר משקל weight. הפעולה תחזיר את מספר הנהגים במערך שהם **נהגי משאית** המורשים להוביל משקל השווה ל-weight או גבוה ממנו.
2. פעולה בשם printAllDrivers (ב-C#: PrintAllDrivers). הפעולה תדפיס את השם ואת סוג הרישיון של כל **נהג שהוא לא נהג משאית**.
3. פעולה בשם printDriversByManager (ב-C#: PrintDriversByManager). הפעולה תדפיס עבור כל מנהל משמרת את שמו ואת שמם של כל נהגי המשאיות בצוות שלו.

(6 נקודות)

6. **מערכת ניהול אנרגיה של שרת מחשוב שולטת בסדרה של רכיבי חומרה. לכל רכיב מוגדר טווח מתח עבודה תקין ומצב נוכחי (פעיל או כבוי). על המערכת לוודא שכל הרכיבים הפעילים עומדים במתח שסופק להם, ולכבות רכיבים שנמצאים בסכנה.**

נתונה המחלקה Component:

Java:

```
public class Component {
 private double low;      // מתח מינימלי
 private double high;     // מתח מקסימלי
 private boolean isActive; // מצב: true - פעיל, false - כבוי
 public Component(double low, double high) {
    this.low = low; this.high = high; this.isActive = true;
 }
 public double getLow() { return this.low; }
 public double getHigh() { return this.high; }
 public boolean getIsActive() { return this.isActive; }
 public void setIsActive(boolean status) {
   this.isActive = status;
 }
}
```

C#:

```
public class Component {
 private double low;      // מתח מינימלי
 private double high;     // מתח מקסימלי
 private bool isActive; // מצב: true - פעיל, false - כבוי
 public Component(double low, double high) {
    this.low = low; this.high = high; this.isActive = true;
 }
 public double GetLow() { return this.low; }
 public double GetHigh() { return this.high; }
 public bool GetIsActive() { return this.isActive; }
 public void SetIsActive(bool status) {
   this.isActive = status;
 }
}
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

א. המערכת בודקת את יציבות הרכיבים תחת מתח כניסה voltage. כתבו פעולה חיצונית המקבלת הפניה לחוליה הראשונה של שרשרת חוליות chain של אובייקטים מסוג Component ומספר ממשי voltage. הפעולה תסרוק את השרשרת **ותכבה** כל רכיב שהמתח voltage **אינו** בטווח התקין שלו. הפעולה תחזיר true אם **לפחות רכיב אחד** נשאר פעיל בתום התהליך, ו-false אם כל הרכיבים כבויים. כותרת הפעולה:

Java:

```
public static boolean filterAndCheck (Node<Component> chain,
                                      double voltage)
```

C#:

```
public static bool FilterAndCheck (Node<Component> chain,
                                   double voltage)
```

(8 נקודות)

ב. המערכת רוצה לבדוק אם ניתן לקבוע **מתח עבודה אחיד** שיאפשר לכל הרכיבים הפעילים כרגע בשרשרת לעבוד בו זמנית. כתבו פעולה חיצונית המקבלת שרשרת חוליות chain. הפעולה תחזיר ערך true אם קיים מתח כלשהו המתאים לכל הרכיבים הפעילים בשרשרת בו-זמנית, ו-false אחרת. כותרת הפעולה:

Java:

```
public static boolean hasCommonVoltage(Node<Component> chain)
```

C#:

```
public static bool HasCommonVoltage(Node<Component> chain)
```

שימו לב: הפעולה לא צריכה למצוא את מתח עבודה אחיד, אלא רק לבדוק אם הוא קיים.

רמז: כדי שמתח כזה יהיה קיים, הערך המינימלי (low) הגבוה ביותר מבין כל הרכיבים __צריך להיות קטן__ מהערך המקסימלי (high) הנמוך ביותר מבין כל הרכיבים.

(8 נקודות)

7.

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

א. כתבו פעולה המקבלת מחסנית של מספרים שלמים st ומספר שלם val, ובודקת אם ערך val נמצא במחסנית לפחות פעמיים.

כותרת הפעולה:

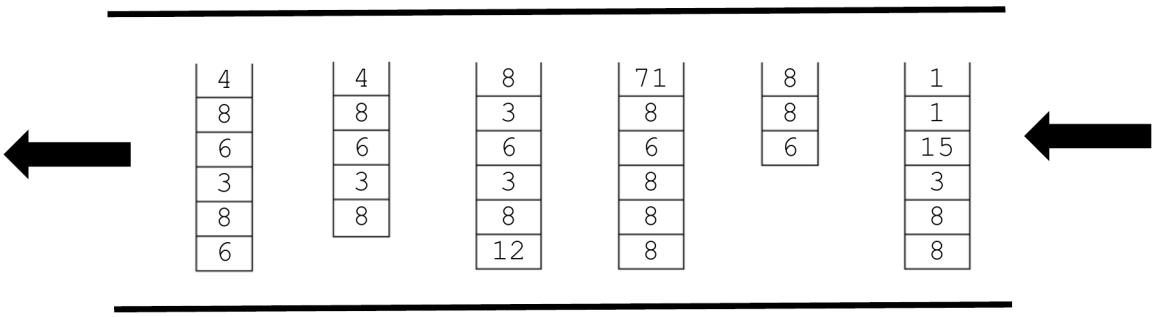
Java:

```
public static boolean isLeastTwice(Stack<Integer> st, int val)
```

C#:

```
public static bool IsLeastTwice(Stack<int> st, int val)
```

(6 נקודות)



ב. נתון תור מחסניות של מספרים שלמים חיוביים. מספר שלם נקרא "ערך זהב" אם הוא מופיע לפחות פעמיים בכל אחת מהמחסניות.

כתבו פעולה המקבלת תור מחסניות של מספרים שלמים que ומחזירה "ערך זהב". אם אין מספר כזה, הפעולה תחזיר ערך (-1).

כותרת הפעולה:

Java:

```
public static int goldValue(Queue<Stack<Integer>> que)
```

C#:

```
public static int GoldValue(Queue<Stack<int>> que)
```

לדוגמה, עבור תור שלפניכם הפעולה תחזיר ערך 8.

(10 נקודות)

ג. מהי הסיבוכיות של הפעולות שכתבתם בסעיפים א' ו-ב'? **הסבירו את תשובתכם.**

(2 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

8. הגדרה: עץ בינארי של מספרים שלמים נקרא "**עץ משלים**" אם הוא עונה על הכללים האלה:

- עבור כל צומת שיש לו **שני בנים**, ערך הצומת שווה ל**סכום** הערכים של שני בניו;
- עבור כל צומת שיש לו **בן יחיד**, ערך הצומת שווה לערכו של הבן היחיד;
- הערכים בכל עלי העץ (צמתים ללא בנים) חייבים להיות מספרים חיוביים (גדולים מ-0).

א. ציירו "עץ משלים" הכולל לפחות שישה צמתים, שבו לפחות לאחד הצמתים יש בן יחיד.

(3 נקודות)

ב. כתבו פעולה חיצונית המקבלת הפניה לשורש של עץ בינארי של מספרים שלמים. הפעולה תבדוק אם העץ הוא "עץ משלים". אם כן - הפעולה תחזיר ערך true, ואם לא - הפעולה תחזיר ערך false. כותרת הפעולה:

Java:

```
public static boolean isComplementary(BinNode<Integer> bt)
```

C#:

```
public static bool IsComplementary(BinNode<int> bt)
```

(12 נקודות)

ג. האם קיים עץ חיפוש בינארי (BST) הכולל לפחות ארבעה צמתים שהוא גם "עץ משלים"? אם כן - ציירו את העץ, אם לא - הסבירו מדוע עץ כזה לא קיים.

(3 נקודות)

9. לפניכם ארבע מחלקות המייצגות היררכיה של מכשירים ניידים והמחלקה Test

Java:

```
class MobileDevice {
   protected int id;
   protected boolean isOn = true;
   protected static int count = 0;

   public MobileDevice () {
      count++;
      this.id = count;
      System.out.print("D" + id + " ");
   }

   public boolean getIsOn(){ return this.isOn; }
   public void togglePower(){ this.isOn = !this.isOn; }

   public String getStatus() {
      if(isOn)return "ID"+id+" [ON]";
      return "ID" + id + " [OFF]";
   }
}// end of MobileDevice

class Phone extends MobileDevice {
   protected boolean isSilent = true;

   public Phone() {  System.out.print("P "); }

   public void setStatus(boolean isOn, boolean isSilent) {
      this.isOn = isOn;
      this.isSilent = isSilent;
   }
   public String getStatus() {
      if(isSilent)return super.getStatus()+" SILENT";
      return super.getStatus() + " RING";
   }
}// end of Phone

class SmartPhone extends Phone {
   public SmartPhone() { System.out.print("S ");}

   public String getStatus() {
      return super.getStatus()+" WiFi" ;
   }
}// end of SmartPhone
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
class Laptop extends MobileDevice {
  public Laptop() { System.out.print("L ");}
}// end of Laptop

public class Test {
  public static void main(String[] args) {
    MobileDevice[] net = new MobileDevice[5];

    System.out.println("--- Step 1: Creation ---");
    net[0] = new SmartPhone();
    System.out.println();
    net[1] = new Laptop();
    System.out.println();
    net[2] = net[0];
    net[3] = new Phone();
    System.out.println();
    net[4] = new MobileDevice();
    System.out.println();

    System.out.println("--- Step 2: Action ---");
    for (int i = 0; i < net.length; i++) {
      if (net[i] instanceof Phone) {
        ((Phone)net[i]).setStatus(true, false);
      } else {
        net[i].togglePower();
      }
    }

    System.out.println("--- Step 3: Final Status ---");
    for (int i = 0; i < net.length; i++)  {
      if (net[i] != null) {
        System.out.println(net[i].getStatus());
      }
    }
  }
}
```

C#:

```
class MobileDevice {
  protected int id;
  protected bool isOn = true;
  protected static int count = 0;

  public MobileDevice () {
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
      count++;
      this.id = count;
      Console.Write("D" + id + " ");
    }

    public bool GetIsOn(){ return this.isOn; }
    public void TogglePower(){ this.isOn = !this.isOn; }

    public virtual string GetStatus() {
      if(isOn)return "ID"+id+" [ON]";
      return "ID" + id + " [OFF]";
    }
}// end of MobileDevice

class Phone : MobileDevice {
    protected bool isSilent = true;

    public Phone() {  Console.Write("P "); }

    public void SetStatus(bool isOn, bool isSilent) {
      this.isOn = isOn;
      this.isSilent = isSilent;
    }
    public override string GetStatus() {
      if(isSilent)return base.GetStatus()+" SILENT";
      return base.GetStatus() + " RING";
    }
}// end of Phone

class SmartPhone : Phone {
    public SmartPhone() { Console.Write("S ");}

    public override string GetStatus() {
      return base.GetStatus()+" WiFi" ;
    }
}// end of SmartPhone

class Laptop : MobileDevice {
    public Laptop() { Console.Write("L ");}
}// end of Laptop

public class Test {
    public static void Main(string[] args) {
      MobileDevice[] net = new MobileDevice[5];

      Console.WriteLine("--- Step 1: Creation ---");
```

```
    net[0] = new SmartPhone();
    Console.WriteLine();
    net[1] = new Laptop();
    Console.WriteLine();
    net[2] = net[0];
    net[3] = new Phone();
    Console.WriteLine();
    net[4] = new MobileDevice();
    Console.WriteLine();

    Console.WriteLine("--- Step 2: Action ---");
    for (int i = 0; i < net.Length; i++) {
      if (net[i] is Phone) {
        ((Phone)net[i]).SetStatus(true, false);
      } else {
        net[i].TogglePower();
      }
    }

    Console.WriteLine("--- Step 3: Final Status ---");
    for (int i = 0; i < net.Length; i++)  {
      if (net[i] != null) {
        Console.WriteLine(net[i].GetStatus());
      }
    }
  }
}
```

א. עקבו אחרי ביצוע קוד ב-main (ב-Main :C#) ורשמו מה יהיה הפלט.

(10 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

ב. מנהלי הפרויקט החליטו להוסיף מעקב אחר **רמת הסוללה** של המכשירים. רמת הסוללה מיוצגת במספר שלם (int) בין 0 ל-100.

רוצים להוסיף תכונה בשם batteryLevel ופעולות getBatteryLevel ו-setBatteryLevel (ב-C#: GetBatteryLevel ו-SetBatteryLevel).

(1) באיזו מחלקה / מחלקות הכי נכון להגדיר את התכונה ואת הפעולות? **נמקו את בחירתכם**.

(2) כתבו פעולה חיצונית (סטטית) בשם **checkCritical** (ב-**CheckCritical**: C#). הפעולה תקבל מערך של מכשירים ניידים ותדפיס את ה-ID של כל המחשבים הניידים (Laptop) במערך שהם במצב **מופעל** ([ON]) ושרמת הסוללה שלהם נמוכה מ-10 (כלומר במצב קריטי).

הערה: הניחו כי הפעולה `getBatteryLevel()` (ב-`GetBatteryLevel()`: C#) כבר קיימת כראוי במחלקות הרלוונטיות.

(8 נקודות)

כל מבחני מה"ט עם פתרונות מלאים: www.guyai.co.il/mahat