

## פרק 12 שאלות בגרות לדוגמה

### 1. מבוא

פרק זה מכיל שלוש שאלות מסכמות על תוכני היחידה. שאלות אלה אמורות לשקף סגנון אפשרי של שאלות המתאים להיכלל בבחינת הבגרות על יחידת הלימוד: עיצוב תוכנה מבוסס עצמים. היקפי השאלות רחבים לעתים מההיקף המומלץ לבחינת בגרות, וכותבי הבחינה יתאימו את היקף השאלה כראוי. מורים יכולים לנצל שאלות אלה במלוא היקפן לצורך תרגול והכנה לבגרות ואף כדפי עבודה במעבדה.

בהצגת שאלות אלה וההנחיות הצמודות אליהן אנו מקווים להשיג כמה מטרות:

1. הצגת דרכים מתאימות לבחינה של תוכני היחידה.
2. הארת האפשרויות השונות הטמונות בשאלות לצורך בחינת נושאים שונים מגוף יחידת הלימוד.
3. הנחיית המורה המכין את כיתתו לבגרות או משתמש בשאלות לסיכום הוראת היחידה, בנושאים הניתנים לפיתוח וסקירה אגב פתרון השאלות.

מובן שסגנונות נוספים של שאלות אפשריים לבחינת הבגרות, אך אנו נתמקד בשאלות המצורפות. שאלות שמפתחים מורים המלמדים את היחידה בפועל, מתפרסמות במאגר השאלות שמפרסם מרכז המורים הארצי למדעי המחשב, ויכולות לשמש כר לדיונים נרחבים ופוריים בין המורים.

ניתן לתת את השאלות כשאלות סגורות במבחן מתכונת, ולדון לאחר מכן בכיווני הפתרון האפשריים ובנושאים המתעוררים בעקבות השאלות. ניתן גם לתת את הדפים כדפי עבודה בכיתה ולפתח את הפתרונות בשלבים תוך התעכבות על הנקודות השונות המצריכות דיון ובידור, על פי הפירוט המצורף לכל שאלה.

## שאלה ראשונה (גרסת ג'אוה)

לפניכם שתי פעולות ו-sod1(...) ו-sod2(...):

```
public static BinTreeNode<Integer> sod1 (int[] a)
{
    bubbleSort(a); //חיון איברי המערך בסדר עולה
    return sod2(a, 0, a.length-1);
}

public static BinTreeNode<Integer> sod2 (int[] a, int left, int right)
{
    int mid = (left+right)/2;

    BinTreeNode<Integer> bt = new BinTreeNode<Integer>(a[mid]);

    if (mid>left)
        bt.setLeft (sod2 (a, left, mid-1));
    if (mid<right)
        bt.setRight (sod2 (a, mid+1, right));

    return bt;
}
```

הפעולה sod1(...) נעזרת בפעולה הבאה, שאותה אין צורך לממש:

```
static void bubbleSort (int[] a)
```

הפעולה ממיינת את איברי המערך בסדר עולה

א. צייר את העץ שיוחזר לאחר הזימון sod1 (a), עבור המערך: a={3,10,-4,20,7,18}.

ב. כתוב את טענת היציאה של הפעולה sod2 (...).

## שאלה ראשונה (גרסת שיטרף)

לפניכם שתי פעולות Sod1(...) ו-Sod2(...):

```
public static BinTreeNode<int> Sod1(int[] a)
{
    BubbleSort(a); //חיון איברי המערך בסדר עולה
    return Sod2(a, 0, a.Length-1);
}

public static BinTreeNode<int> Sod2(int[] a, int left, int right)
{
    int mid = (left+right)/2;

    BinTreeNode<int> bt = new BinTreeNode<int>(a[mid]);

    if (mid>left)
        bt.SetLeft(Sod2(a, left, mid-1));
    if (mid<right)
        bt.SetRight(Sod2(a, mid+1, right));

    return bt;
}
```

הפעולה Sod1(...) נעזרת בפעולה הבאה, שאותה אין צורך לממש:

```
static void BubbleSort(int[] a)
```

הפעולה ממיינת את איברי המערך בסדר עולה

ג. צייר את העץ שיוחזר לאחר הזימון Sod1(a), עבור המערך: a={3,10,-4,20,7,18}.

ד. כתוב את טענת היציאה של הפעולה Sod2(...).

## נושאים עיקריים שנבחנו בעזרת השאלה:

א. מעקב אחר קוד רקורסיבי (רקורסיה לא פשוטה).

ב. שימוש בפעולות ממשק.

ג. שימוש אלגוריתמי באוספים ובטיפוסי נתונים שנלמדו ביחידה לצורך פתרון בעיות.

ד. תרגול עצים בינריים ובנייה של עץ חיפוש בינרי בפרט.

ה. הגדרת משימה אלגוריתמית מתוך קוד נתון.

הערה: השאלה אינה כוללת סיפור מפורט ממנו צריך התלמיד לברור את הפרטים הנחוצים.

## כיווני פיתוח אפשריים נוספים

1. כתוב פעולה המקבלת מספר שלם  $n$  כפרמטר ובונה עץ בינרי מלא או כמעט מלא שבו  $n$  צמתים.

הערה: בנוסח זה רצוי לציין במפורש את כותרת הפעולה:

```
public BinTreeNode<Integer> buildFullTree (int n) : ג'אוה:  
public BinTreeNode<int> BuildFullTree (int n) : ס'שרפ:
```

2. ניתן להמשיך את השאלה מסעיף 1 ולבקש לכתוב קוד של פעולה המקבלת רשימה או מערך של  $n$  מספרים שלמים שונים זה מזה, הממוינים בסדר עולה. הפעולה מאחסנת את איברי הסדרה הממוינים בעץ שנבנה בסעיף 1 תוך שמירה על הסדר שלהם.

3. אפשרות שונה: לתת את סעיף 1 ולאחר מכן את הקוד **המבצע** את סעיף 2 ולבקש לעקוב ולהגדיר את טענת היציאה של הפעולה.

4. כיוון נוסף: לתת לתלמידים קוד של פעולה המקבלת שני פרמטרים:

א. הפניה לחוליה בינרית המשמשת למעשה עץ בינרי כמעט מלא שבו  $n$  צמתים.

ב. רשימה ממוינת של  $n$  מספרים שלמים השונים זה מזה.

הפעולה ממקמת את ערכי הרשימה במבנה העץ הקיים כך שהסדר הממוין של הסדרה נשמר בעץ.

### נוסח קודם של השאלה (הופיע במהדורה הראשונה של המדריך למורה)

הנוסח המקורי של השאלה כפי שהוא מופיע להלן, הסתבר כמורכב יותר וקשה מדי לפתרון על ידי התלמיד בזמן מבחן. אי לכך פורקה השאלה ועוצבה בכמה אופנים המפורטים לעיל. ניתן להשתמש בנוסח המקורי כדף מעבדה ותרגול לקראת הבגרות.

כתבו פעולה המקבלת רשימה לא-ריקה, ממוינת, של מספרים שלמים שונים זה מזה, ומחזירה עץ-חיפוש-בינרי מלא (או כמעט מלא) שבו מופיעים כל המספרים שברשימה.

#### הנחיות לפתרון:

- א. בנו עץ בינרי מלא (או כמעט מלא) שמספר הצמתים בו כמספר איברי הרשימה.
- ב. עברו על העץ בסריקה בסדר תוכי ועדכנו את ערכי צמתיו במספרים השמורים ברשימה, מתחילתה ועד סופה.

#### נושאים עיקריים שנבחנו בעזרת השאלה:

- א. רשימה, ובפרט רשימה ממוינת
- ב. עץ בינרי, עץ בינרי מלא, סריקות על עצים

#### שלבים בפתרון:

1. החליטו אם לשלב את ההנחיות לפתרון בגוף השאלה עצמה. הדבר עשוי למנוע הסתבכויות ומשאיר עדיין מרחב מספיק לתלמידים להוכיח את שליטתם בחומר.
2. יש לבנות רשימה (אקראית) של מספרים שלמים ולמיינה. תוך כדי כך יש למנות את מספר הערכים שברשימה (מספר הצמתים נחוץ לצורך בניית העץ).
3. יש לבנות עץ בינרי מלא או כמעט מלא, תלוי במספר הערכים שברשימה. כדי לוודא שהעץ נבנה בצורה הנדרשת יש להשתמש בתור ובתבנית הסריקה לפי רמות. ראשית נבנה שורש העץ. כל זמן שעוד לא נבנו צמתים במספר הערכים שברשימה בדיוק, יש לשלוף צומת מהתור, לבנות לו תת-עץ או שניים על פי הצורך, ולהכניס את הצמתים החדשים לתוך התור. יש להחליט על ערך שיאוחסן בצומתי העץ בשלב הבנייה. לערך זה כמובן אין שום משמעות.
4. בשלב זה יש לבצע סריקה בסדר תוכי של העץ, ולהכניס אליו את הערכים השמורים ברשימה. הערך הראשון מהרשימה (זהו הערך הקטן ביותר שבה), יוכנס לעלה השמאלי ביותר בעץ. תהליך הכנסת הערכים לעץ יכול: הוצאת ערך מראש הרשימה והכנסתו למקום המתאים במהלך הסריקה. זוהי למעשה הכנסת איברים לעץ-חיפוש-בינרי כאשר העץ קיים ולא נבנה תוך כדי ההכנסה. ניתן להסתכל על הפעולה גם כפעולה הפוכה לפעולת המיון בעזרת עץ-חיפוש-בינרי המופיעה בפרק.
5. הערה: הפתרונות שבאתר נעזרים בפעולות עזר שונות הקיימות בספריות העזר. על התלמיד לממש כל פעולה שהוא משתמש בה. הפתרון כולל גם פעולות שהתלמיד אינו נדרש לבצע (הדפסת העץ למשל).

### נקודות לדיון:

1. לא ניתן לבצע מיון בעזרת עץ כפי שהוא מופיע בפרק, שכן אז אין דרך להשתלט על מבנה העץ כך שיהיה מלא או כמעט מלא. לצורך כך חייבים לבנות קודם כול את העץ עצמו.

## שאלה שנייה (גרסת ג'אוה)

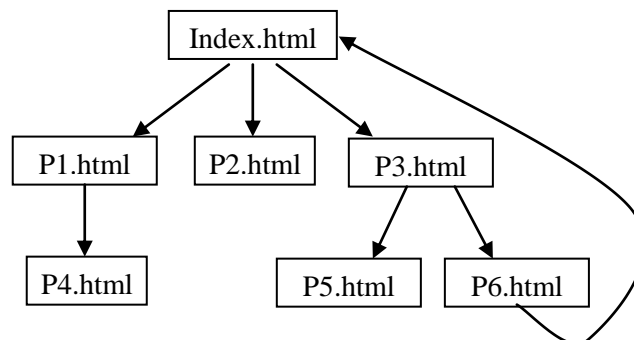
דף אינטרנט מכיל בתוכו טקסטים, תמונות, קישוריות ורכיבים נוספים. בשאלה זו נתמקד בדף אינטרנט המכיל רק קישוריות (לינקים) לדפי אינטרנט אחרים. קישוריות היא הפניה לדף. מספר הקישוריות שדף יכול להכיל אינו מוגבל.

### ממשק המחלקה Page

המחלקה מגדירה את הטיפוס דף אינטרנט

Page (String name)	הפעולה בונה דף ריק (ללא קישורים) ששמו מתקבל כפרמטר
String getName()	הפעולה מחזירה את שם הדף
void addLink (Page link)	הפעולה מוסיפה לדף הנוכחי קישור לדף המתקבל כפרמטר
Page[] getAllLinks()	הפעולה מחזירה מערך המכיל את כל הקישורים הנמצאים בדף הנוכחי. אם הדף הנוכחי ריק, גודל המערך שיוחזר יהיה 0

א. אתר אינטרנט בנוי מדפי אינטרנט המקושרים ביניהם. כתבו קוד היוצר את מבנה דפי האינטרנט כמתואר בצירור:



ב. כתבו את המחלקה Page.

ג. ממשו את הפעולה הפנימית. לצורך כך, הניחו שההפניות בין הדפים אינן מייצרות מעגלים מסוג המעגל המתואר באיור שבסעיף א:

```
public boolean canGoTo (Page page)
```

הפעולה מחזירה 'אמת' אם ניתן להגיע מהדף הנוכחי לדף המתקבל כפרמטר, ו-'שקר' אחרת.

## שאלה שנייה (גרסת שיורפ)

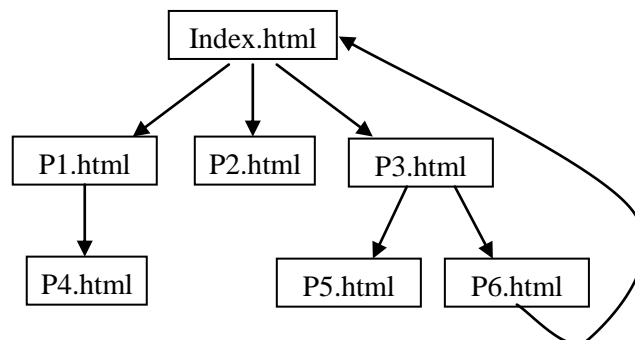
דף אינטרנט מכיל בתוכו טקסטים, תמונות, קישוריות ורכיבים נוספים. בשאלה זו נתמקד בדף אינטרנט המכיל רק קישוריות (לינקים) לדפי אינטרנט אחרים. קישוריות היא הפניה לדף. מספר הקישוריות שדף יכול להכיל אינו מוגבל.

### ממשק המחלקה Page

המחלקה מגדירה את הטיפוס דף אינטרנט

Page (string name)	הפעולה בונה דף ריק (ללא קישורים) ששמו מתקבל כפרמטר
string GetName()	הפעולה מחזירה את שם הדף
void AddLink (Page link)	הפעולה מוסיפה לדף הנוכחי קישור לדף המתקבל כפרמטר
Page[] GetAllLinks()	הפעולה מחזירה מערך המכיל את כל הקישורים הנמצאים בדף הנוכחי. אם הדף הנוכחי ריק גודל המערך שיוחזר יהיה 0

א. אתר אינטרנט בנוי מדפי אינטרנט המקושרים ביניהם. כתבו קוד היוצר את מבנה דפי האינטרנט כמתואר בציור:



ב. כתבו את המחלקה Page.

ג. ממשו את הפעולה הפנימית. לצורך כך, הניחו שההפניות בין הדפים אינן מייצרות מעגלים מסוג המעגל המתואר באיור שבסעיף א:

```
public bool CanGoTo (Page page)
```

הפעולה מחזירה 'אמת' אם ניתן להגיע מהדף הנוכחי לדף המתקבל כפרמטר, ו-'שקר' אחרת.



**הצעה חלופית לניסוח סעיף ג שבשאלה:**

- ניתן לנסח את סעיף ג בצורה שונה ולבקש לכתוב פעולה **חיצונית** המקבלת **שני דפים** ומחזירה 'אמת' אם ניתן להגיע מהדף הראשון אל הדף השני, ו-'שקר' אחרת.
- שימו לב כי כאשר מבקשים פעולה פנימית, יש לשלוח לפעולות את הפרמטר **this** המייצג את הדף הנוכחי שבו עוסקים. התלמידים פגשו מעט מאוד העברת פרמטרים שכזו: ג'אוה:

```
public boolean canGoTo (Page page)
{
    Queue<Page> q = new Queue<Page> ();
    q.insert (this);
    while (!q.isEmpty())
    {...}
}
```

סישרפ:

```
public bool CanGoTo (Page page)
{
    Queue<Page> q = new Queue<Page> ();
    q.Insert (this);
    while (!q.IsEmpty())
    {...}
}
```

- גם המורים שישנו את נוסח הסעיף ויבקשו לכתוב פעולה חיצונית, מוזמנים להציג לפני תלמידיהם את האפשרות של כתיבת הפעולה כפעולה פנימית.

**כיווני ניסוח אפשריים**

- השאלה כלשונה מעט עמוסה מדי. להלן כמה הצעות שיפור אפשריות:
- ניתן לוותר בזמן מבחן על הדרישה של סעיף ג ולהסתפק בשני הסעיפים הראשונים.
  - ניתן להמיר בסעיף ג את כתיבת הקוד למעקב אחר קוד. גיוון שכזה יאפשר גם לתלמידים שמעדיפים את המיומנות של מעקב אחר קוד לבחור בשאלה.

**הערות:**

- הדרישה "כתוב מחלקה" המופיעה בשאלה זו כמו בשאלות נוספות, כוללת בחובה את הנוסח הישן של "בחר ייצוג", "ייצג" ולאחר מכן את שלב המימוש על פי הייצוג. הדרישה לכתוב את המחלקה כולה, מחייבת את התלמיד לבחור בייצוג מסוים ולהתייחס לייצוג בעת מימוש הפעולות. מכיוון שנדרש מימוש מלא של המחלקה לא יכול התלמיד ל"התחכם" ולבחור בייצוג שיועיל לו לצורך ספציפי מסוים ועליו להציג ייצוג טוב שיענה על צרכי כלל הפעולות של המחלקה.

- הפעולה `addLink(...)` / `AddLink(...)` אינה מגדירה סדר בין הקישוריות שבדף, ולכן סביר יותר להשתמש בפעולה החסכונית `insert(null, ...)` / `Insert(null, ...)` – המכניסה כל דף לראש רשימת הדפים.
- הציור של דפי האינטרנט עשוי לייצר ברגע הראשון תחושה של עץ כללי, בו לכל חוליה מספר לא מוגבל של ילדים ושבו מותרים מעגלים בין הצמתים. יש להסביר ולדייק שלמרות התחושה הראשונה שמדובר בעץ, למעשה המבנה אינו עומד בהגדרת העץ כפי שנוסחה ביחידה. בכל זאת אין מניעה לייצג את המבנה של דף אינטרנט בעזרת רשימות.
- הפעולה `getAllLinks()` / `GetAllLinks(...)` מחזירה מערך שגודלו נקבע ברגע זימון הפעולה. חשוב שהתלמידים יבינו שהמערך נוצר כאשר מספר הקישוריות ידוע, ולכן הוא בדיוק בגודל המתאים.
- השאלה מעוצבת אמנם כשאלת סיפור אך הפרטים בה מעטים ואינם מכבידים למצוא את העיקר אליו צריך להתייחס, מה גם שהסיפור מציג שימוש אמיתי ומתקבל על הדעת של הנושאים שנלמדו ביחידה.

#### הנחיות לפתרון:

1. בסעיף א יש לשים לב שהתלמידים מבינים שאין צורך לכתוב תוכנית כללית כלשהי, אלא יש לכתוב קוד המייצר את המצב המתואר בלבד. בסעיף א נדרשים התלמידים לבנות מעגל בין החוליות. בסעיף ג כדי למנוע מצב של לולאה אין סופית מתבקשים התלמידים להניח שההפניות אינן יוצרות מעגלים.
2. לצורך פתרון סעיף ג צריכים התלמידים לנצל את האלגוריתם לסריקת עץ לפי רמות (בעזרת תור). לתור זה מכניס התלמיד את הדף הנוכחי. ומכאן והלאה הוא משחרר בכל פעם דף מראש התור ומכניס לסופו את הדפים שברשימה השמורה בו, תוך בדיקה אם הם אינם כוללים את דף היעד שהוא מעוניין להגיע אליו.

#### נושאים עיקריים שנבחנו בעזרת השאלה:

- שאלה זו מציגה טיפול באוסף בעזרת מבנה חוליות ומתרגלת שימוש במבנים שנלמדו לצורך שמירת אוסף וסריקתו.
- השאלה מתרגלת שימוש נכון בהפניות, שימוש בתור ושימוש ברשימה.
- השאלה מתרגלת אלגוריתמיקה גם ללא הצורך בכתיבת אלגוריתם במפורש. סעיף ב מחייב חשיבה אלגוריתמית לתכנון דרך הפתרון של המשימה.

#### נקודות לדין ולמחשבה:

1. השאלה מדברת למעשה על מבנה של גרף: לצומת יש מספר לא מוגבל של ילדים ויכולים להיווצר מעגלים בין הצמתים. למרות שלצורך התכנות של סעיף ג יש הסתייגות מקיום מעגלים וניתן היה להגדיר שהשאלה מתמקדת בעץ כללי, הרי למעשה אין מניעה להשאיר את

האפשרות של קיום מעגלים ולהוסיף אינדיקציה מתאימה שתבדוק אם הגענו כבר לדף קיים. אם נמצא שחזרנו לדף שבו כבר ביקרנו (למשל `Index.html` שבאיור), נימנע מלהמשיך את החיפוש בנתיב זה וכך נימנע מלולאה אין סופית. האינדיקציה יכולה להיות תכונה נוספת (בוליאנית) בדף עצמו, אלא שאין בכך היגיון שהרי הדף מיועד עבור יישומים שונים שבהם אין לתכונה זו שום משמעות. האינדיקציה יכולה להיות בעזרת מפה נלווית שתצמיד לכל דף שאליו נגיע (המפתח במפה) ערך שיציין שכבר הגענו לדף זה. השימוש במפה יכול להיות מעניין כשלעצמו. שימו לב ששם הדף יכול לשמש כמפתח שכן הוא ייחודי.

### שאלה 3 (גרסת ג'אוה)

במטרה לקבוע מי הם אתרי האינטרנט האהובים ביותר, נבנתה מערכת ממוחשבת הקולטת מגולשים את הדירוג לאתרי אינטרנט שהם אוהבים. הגולשים יכולים לדרג מספר לא מוגבל של אתרים. הדירוג נעשה על ידי מתן ניקוד שלם בין 1 ל-10. המערכת הממוחשבת מנהלת "רשימת רייטינג" שבה נשמרות כתובות האתרים שדורגו והניקוד המצטבר שלהם. רשימת הרייטינג מתעדכנת ונשמרת בצורה ממוינת בסדר יורד על פי הניקוד המצטבר (האתר עם הניקוד הגבוה ביותר נמצא בתחילת הרשימה), כך ניתן לדעת בכל רגע, מהם האתרים האהובים ביותר.

המערכת הממוחשבת נעזרת בשתי מחלקות לביצוע המשימה:

1. המחלקה SiteRating המגדירה זוג ערכים: כתובת האתר ומספר הנקודות המצטבר שקיבל. להלן ממשק המחלקה:

SiteRating (String url, int points)	הפעולה בונה זוג ערכים: כתובת אתר האינטרנט והניקוד ההתחלתי שקיבל
String getURL()	הפעולה מחזירה את כתובת האתר הנוכחי
int getPoints()	הפעולה מחזירה את מספר הנקודות המצטבר של האתר הנוכחי
void addPoints (int points)	הפעולה מוסיפה ניקוד למספר הנקודות המצטבר
String toString()	הפעולה מתארת כתובת האתר ומספר הנקודות המצטבר שקיבל

2. המחלקה RatingList המנהלת את "רשימת הרייטינג":

```
public class RatingList
{
    private List<SiteRating> ratingList; // רשימת הרייטינג
    // פעולה הבונה רשימת רייטינג ריקה
    public RatingList ()
    {
        this.ratingList = new List<SiteRating> ();
    }
    // הפעולה מדרגת אתר על ידי מתן ניקוד ומעדכנת את רשימת הרייטינג על פי הניקוד המצטבר
    public void rate(String url, int points)
    {
        // השלימו ...
    }
    // הפעולה מתארת את רשימת הרייטינג בצורה ממוינת, בסדר יורד, על פי הניקוד המצטבר
    public String toString()
    {
        return this.ratingList.toString();
    }
}
```

ממשו את הפעולה rate(...) החסרה במחלקה RatingList. הפעולה מדרגת אתר שכתובתו url:

- אם האתר דורג בעבר, הפעולה תוסיף לניקוד המצטבר שלו points נקודות.
- אם זו הפעם הראשונה שהאתר מדורג, הפעולה תוסיף את האתר החדש והניקוד שקיבל לרשימת הרייטינג.
- בשני המצבים, הפעולה תעדכן את רשימת הרייטינג כך שתישאר ממוינת בסדר יורד על פי הניקוד המצטבר של האתרים.

לצורך הבהרת המשימה, ניתן להוסיף לצד השאלה את הדוגמה הבאה:  
לאחר שנקלטו הדירוגים האלה:

```
RatingList rl = new RatingList();  
  
rl.rate("www.google.com", 9);  
rl.rate("www.ynet.co.il", 2);  
rl.rate("www.ynet.co.il", 8);  
rl.rate("www.google.com", 4);  
rl.rate("www.walla.co.il", 7);  
rl.rate("www.ynet.co.il", 1);  
rl.rate("www.google.com", 10);  
rl.rate("www.ynet.co.il", 9);  
  
System.out.println(rl);
```

היה מצב רשימת הדירוג כזה:

```
[www.google.com=23, www.ynet.co.il=20, www.walla.co.il=7]
```

### שאלה 3 (גרסת סירפ)

במטרה לקבוע מי הם אתרי האינטרנט האהובים ביותר, נבנתה מערכת ממוחשבת הקולטת מגולשים את הדירוג לאתרי אינטרנט שהם אוהבים. הגולשים יכולים לדרג מספר לא מוגבל של אתרים. הדירוג נעשה על ידי מתן ניקוד שלם בין 1 ל-10. המערכת הממוחשבת מנהלת "רשימת רייטינג" שבה נשמרות כתובות האתרים שדורגו והניקוד המצטבר שלהם. רשימת הרייטינג מתעדכנת ונשמרת בצורה ממוינת בסדר יורד על פי הניקוד המצטבר (האתר עם הניקוד הגבוה ביותר נמצא בתחילת הרשימה), כך ניתן לדעת בכל רגע, מהם האתרים האהובים ביותר.

המערכת הממוחשבת נעזרת בשתי מחלקות לביצוע המשימה:

1. המחלקה SiteRating המגדירה זוג ערכים: כתובת האתר ומספר הנקודות המצטבר שקיבל. להלן ממשק המחלקה:

SiteRating (string url, int points)	הפעולה בונה זוג ערכים: כתובת אתר האינטרנט והניקוד ההתחלתי שקיבל
string GetURL()	הפעולה מחזירה את כתובת האתר הנוכחי
int GetPoints()	הפעולה מחזירה את מספר הנקודות המצטבר של האתר הנוכחי
void AddPoints (int points)	הפעולה מוסיפה ניקוד למספר הנקודות המצטבר
string ToString()	הפעולה מתארת כתובת האתר ומספר הנקודות המצטבר שקיבל

2. המחלקה RatingList המנהלת את "רשימת הרייטינג":

```
public class RatingList
{
    private List<SiteRating> ratingList; // רשימת הרייטינג
    // פעולה הבונה רשימת רייטינג ריקה
    public RatingList ()
    {
        this.ratingList = new List<SiteRating> ();
    }
    // הפעולה מדרגת אתר על ידי מתן ניקוד ומעדכנת את רשימת הרייטינג על פי הניקוד המצטבר
    public void Rate(string url, int points)
    {
        // השלימו ...
    }
    // הפעולה מתארת את רשימת הרייטינג בצורה ממוינת, בסדר יורד, על פי הניקוד המצטבר
    public override String ToString()
    {
        return this.ratingList.ToString();
    }
}
```

ממשו את הפעולה Rate(...) החסרה במחלקה RatingList. הפעולה מדרגת אתר שכתובתו url:

- אם האתר דורג בעבר, הפעולה תוסיף לניקוד המצטבר שלו points נקודות.
- אם זו הפעם הראשונה שהאתר מדורג, הפעולה תוסיף את האתר החדש והניקוד שקיבל לרשימת הרייטינג.
- בשני המצבים, הפעולה תעדכן את רשימת הרייטינג כך שתישאר ממוינת בסדר יורד על פי הניקוד המצטבר של האתרים.

ניתן להוסיף לצד השאלה את הדוגמה הבאה לצורך הבהרת המשימה:  
לאחר שנקלטו הדירוגים האלה:

```
RatingList rl = new RatingList();  
  
rl.Rate("www.google.com", 9);  
rl.Rate("www.ynet.co.il", 2);  
rl.Rate("www.ynet.co.il", 8);  
rl.Rate("www.google.com", 4);  
rl.Rate("www.walla.co.il", 7);  
rl.Rate("www.ynet.co.il", 1);  
rl.Rate("www.google.com", 10);  
rl.Rate("www.ynet.co.il", 9);  
  
Console.WriteLine(rl);
```

היה מצב רשימת הדירוג כזה:

```
[www.google.com=23, www.ynet.co.il=20, www.walla.co.il=7]
```

### כיווני ניסוח אפשריים

ניתן להקדים שימוש במחלקה למימושה, ברוח המבנה של היחידה: להציג בסעיף א, את הדוגמה שלעיל ולבקש מהתלמיד לכתוב איך תראה רשימת הדירוג. רק לאחר השימוש, יפנה התלמיד למימוש הפעולה המבוקשת.

### נושאים עיקריים שנבחנו בעזרת השאלה:

- עבודה עם ממשקים.
- פעולות מיון מתקדמות.
- עבודה עם טיפוס אוסף (רשימה).
- השלמת קוד.

### הערה:

הפעם הייצוג במחלקה RatingList הוא ייצוג נתון והתלמיד חייב להתאים את עצמו אליו.

### נקודות לדיון ולמחשבה:

- האם הרשימה היא טיפוס האוסף היחיד המתאים למימוש רשימת הדירוג? תלמידים שמכירים את טיפוס המפה יכולים לבחור להשתמש בו. כאן יתאים לנהל דיון לגבי השמירה של האיברים במפה. האם איברים במפה נשמרים על פי סדר כלשהו? מכיוון שסיפור המעטפת מחייב מיון לפי הערך השמור במפה בעוד הזיהוי נעשה לפי המפתח, המפה אינה אוסף המתאים לשאלה הנוכחית.
- ניתן לוותר על דרישת המיון ולאפשר פתרון של השאלה בעזרת מפה (יתאים אולי בשנים הבאות). שימוש במפה מצריך הגדרת טיפוס חדש של זוג.

