

חומרים שהוכנו על-ידי משתתפי קורס מורים מובילים

תשע"ה

ניתן להשתמש בחומרים לצורך הוראה בלבד.

לא ניתן לפרסם את החומרים או לעשות בהם כל שימוש מסחרי

ללא קבלת אישור מראש מצוות הפיתוח

כתיבה ועריכה:

ולרי פקר

דפים מדורגים לתלמידי "עיצוב תוכנה"

כתיבת פעולה רקורסיבית לפתרון החידה " מגדלי האנוי " וניתוח זמן הריצה של הפעולה.

מדריך למורה:

רקע: כתיבת פעולות רקורסיביות נחשבת כחומר קשה להוראה. המעבדה מבוססת על שיטת פירוק *Unpacking* של בעיה רקורסיבית, המאפשרת לפשט את מורכבות כתיבת הפעולה. לפי השיטה, מתחילים בבדיקת מקרים פשוטים לפתרון הבעיה ולא בכתיבת הפעולה. בדיקת הפתרונות נעשית בצורה הדרגתית ממקרה פשוט למקרה מורכב, בעזרת האפליקציה הגרפית שכתבתי עבור המעבדה. בשלב הראשון, תלמידים מריצים את האפליקציה ופותרים את החידה עבור כמה מיקרים ומחפשים מכנה משותף במעבר בין המקרים (מפשוט למורכב). בשלב השני, תלמידים עונים על השאלה: "האם קיימת אסטרטגיה לפתרון הבעיה"? האם פתרון הבעיה עבור מקרה פשוט יכול לעזור בפתרון הבעיה המורכבת יותר? האם כדי לפתור בעיה או חלק כלשהו ממנה, יש לפתור חלק קטן יותר (או חלקים קטנים יותר) של אותה בעיה? בשלב השלישי, תלמידים מנסים לתאר מילולית את פתרון הבעיה הכללית בצורה רקורסיבית. רק אז מתחילים בהמרת התיאור המילולי לקוד בשפת C#. בשלב האחרון, תלמידים מתחילים משימת חקר להשוואה בין זמנים "אמיתיים" של הפעולה הרקורסיבית על אורכי קלט שונים (מספר דסקיות שונה).

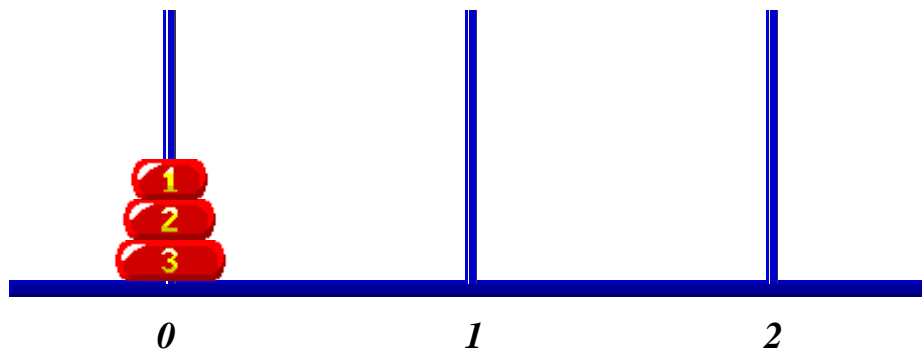
כתיבת הקוד מבוססת על הקובץ - *Unit4.dll*, המכיל, בין היתר, את המחלקה הגרפית *Tower*. המחלקה מאפשרת לבנות ולהציג מוט או מספר מוטות מטיפוס המחלקה. בשלב בניית המוט נקבעים ערכי שתי התכונות: כמות הדסקיות שמושחלות על המוט וכותרתו. המחלקה הגרפית מציגה את כל מופע של מוט באופן ויזואלי על החלון הגרפי ומצרפת את הכותרת לכל המופע. הייצוג הויזואלי מפשט לתלמיד את תהליך העבודה ומקל עליו את התחלת העבודה.

המעבדה בנויה מסדרת דפים, שכל אחד מהם נותן לתלמיד רמז, המסייע לו באופן מדורג בפתרון המשימה. המורה מחלק לתלמידים דף (מספר 1) ועליו תיאור של משימה מורכבת שעליהם לבצע. על שולחן המורה נמצאים דפים נוספים: דף מספר 2, דף מספר 3, וכן הלאה. בכל דף כזה ניתנים לתלמיד רמזים קלים שיכולים לקדם אותו מעט בביצוע המשימה. (לא פתרון מלא של כל המשימה, אלא רק דחיפה בכיוון המתאים). השיעור מתנהל כך שכל התלמידים מקבלים את הדף הראשון אבל כל תלמיד מחליט באופן עצמאי מתי (ואם בכלל) הוא ניגש לשולחן ולוקח את הדף הבא. מותר לתלמיד לקחת גם מספר דפים בו-זמנית.

דף מספר 1

מגדלי הנוי - *Towers of Hanoi*, היא חידה מתמטית שהומצאה על-ידי המתמטיקאי הצרפתי אדוארד לוקאס ב-1883 והפכה למשחק. המשחק בנוי משלושה מוטות. בתחילת המשחק, על מוט אחד מושחלות n דסקיות בסדר יורד ביחס להיקפן (כמתואר באיור), היוצרות צורה של מגדל - ומכאן שמו של המשחק. מטרת החידה להעביר את מגדל הדסקיות בשלמותו מהמוט המלא 0 , למוט הריק 2 , תוך שימוש במוט 1 , על פי הכללים הבאים:

- בכל שלב, מותר להעביר דסקית אחת בלבד ממוט למוט.
- באף שלב אסור שדסקית גדולה תהיה מונחה על דסקית קטנה ממנה.



- ✓ לצורך המשימה הכנתי עבורכם אפליקציה לסימולציה של החידה. האפליקציה מאפשרת להעביר דסקית אחת באמצעות העכבר ממוט למוט.
- הריצו את האפליקציה ופתרו את החידה עבור 2 דסקיות.
 - פתרו את החידה עבור 3 דסקיות.
- האם אפשר להיעזר בפתרון החידה עבור 2 דסקיות כדי לפתור את החידה עבור 3 דסקיות? הסבירו.

- פתרו את החידה עבור 4 דסקיות.
- האם אפשר להיעזר בפתרון החידה עבור 3 דסקיות כדי לפתור את החידה עבור 4 דסקיות? הסבירו.

- האם קיימת אסטרטגיה לפתרון החידה? כלומר... האם תוכלו לנסח פתרון כללי?

דף מספר 2

לפניכם פיתרון המשימה המופיעה בדף 1.

✓ פתרון החידה עבור 3 דסקיות:

בהסתמך על פתרון החידה עבור 2 דסקיות, נוכל למצוא את פתרון החידה עבור 3 דסקיות.

- נעביר את שתי הדסקיות העליונות 1, 2 ממוט 0 למוט 1, תוך שימוש במוט 2.
- נעביר את הדסקית 3 ממוט 0 למוט 2.
- נעביר את שתי הדסקיות 1, 2 ממוט 1 למוט 2, תוך שימוש במוט 0.

✓ פתרון החידה עבור 4 דסקיות:

בהסתמך על פתרון החידה עבור 3 דסקיות, נוכל למצוא את פתרון החידה עבור 4 דסקיות.

- נעביר את שלוש הדסקיות העליונות 1, 2, 3 ממוט 0 למוט 1, תוך שימוש במוט 2.
- נעביר את הדסקית 4 ממוט 0 למוט 2.
- נעביר את שלוש הדסקיות 1, 2, 3 ממוט 1 למוט 2, תוך שימוש במוט 0.

✓ פתרון החידה עבור n דסקיות:

בהסתמך על פתרון החידה עבור $n-1$ דסקיות, נוכל למצוא את פתרון החידה עבור n דסקיות.

- נעביר את $n-1$ הדסקיות העליונות 1, 2, 3, ..., $n-1$ ממוט 0 למוט 1, תוך שימוש במוט 2.
- נעביר את הדסקית n ממוט 0 למוט 2.
- נעביר את $n-1$ הדסקיות 1, 2, 3, ..., $n-1$ ממוט 1 למוט 2, תוך שימוש במוט 0.

כדי לפתור בעיה או חלק כלשהו ממנה, יש לפתור חלק קטן יותר (או חלקים קטנים יותר) של אותה בעיה.

לפניכם התיאור הרקורסיבי לפיתרון החידה:

העבר (n) דסקיות ממוט מקור למוט יעד

- ✓ העבר $n - 1$ דסקיות ממוט המקור למוט הנוסף
- ✓ העבר דסקית אחת ממוט המקור למוט היעד
- ✓ העבר $n - 1$ דסקיות מהמוט הנוסף למוט היעד

דף מספר 3

לרשותכם מחלקה גרפית בשם *Tower*. כל מוט מטיפוס המחלקה מתאפיין בכמות דסקיות המושחלות עליו ובכותרת. הדסקיות שמושחלות על המוט מסודרות בסדר יורד ביחס להיקפן. המחלקה מופיעה תחת מרחב השמות *Unit4.TowerLib*. לצורך שימוש במחלקה מוט עליכם לצרף את הקובץ *Unit4.dll* לפרויקט ולהצהיר על מרחב השמות בראש התוכנית.

לפניכם ממשק המחלקה הגרפית:

שם הפעולה	תאור הפעולה
<i>Tower(string caption)</i>	הפעולה בונה מוט ריק שכותרתו <i>caption</i> .
<i>Tower(int n, string caption)</i>	הפעולה בונה מוט. על מוט מושחלות <i>n</i> דסקיות בסדר יורד ביחס להיקפן. כותרתו של המוט היא <i>caption</i> . <u>הנחה</u> : $0 \leq n \leq 9$
<i>void MoveDisk(toTower)</i>	הפעולה מעבירה את הדסקית העליונה מהמוט הנוכחי למוט <i>toTower</i> . הפעולה מעבירה את הדסקית רק בצורה תקינה (דסקית גדולה לא תהיה על דסקית קטנה, והמוט הנוכחי לא ריק).

✓ כתבו תוכנית המדמה את משחק "מגדלי הנווי". התכנית תיצור מערך של שלושה מוטות. בתחילת המשחק, על המוט במקום הראשון במערך מושחלות 5 דסקיות ושני המוטות האחרים ריקים. מטרת המשחק היא להעביר את מגדל הדסקיות בשלמותו מהמוט במקום 0 למוט ביעד 2.

✓ התוכנית תעביר את מגדל הדסקיות בשלמותו מהמוט במקום 0 למוט במקום 2 בעזרת הפעולה הרקורסיבית:

private static void HanoiTowersSolution(Tower[] towers, int nRings, int fromPole, int toPole)

הפעולה מקבלת מערך של שלושה מוטות *towers* ומספר דסקיות *nRings* להעברה ממוט במקום *fromPole* (מוט המקור) למוט במקום *toPole* (מוט היעד).

דף מספר 4

לפניכם המחלקה הראשית לפתרון המשחק :

```
using System;
using Unit4.TowerLib;
namespace TestHanoiTower
{
    class HanoiTower
    {
        static void Main(string[] args)
        {
            // שלב ראשון – יצירת עצמים שמייצגים את משחק מגדלי הנוי
            Tower[] towers = new Tower[3];
            towers[0] = new Tower(5, "0");
            towers[1] = new Tower("1");
            towers[2] = new Tower("2");
            // שלב שני – זימון הפעולות לפתרון משחק מגדלי הנוי
            HanoiTowersSolution(towers , 5, 0, 2);
        }
        // פעולה לפתרון משחק מגדלי הנוי
        private static void HanoiTowersSolution(Tower[] towers, int nRings,
            int fromPole, int toPole)
        {
            int extraPole;
            extraPole = // כאן צריך לחשב את מספרו של המוט הנוסף
        }
        // כאן צריך להעביר  $n - 1$  דסקיות ממוט המקור למוט הנוסף
        towers[fromPole].MoveDisc(towers[toPole]);
        // כאן צריך להעביר  $n - 1$  דסקיות מהמוט הנוסף למוט היעד
    }
}
}
```

✓ השלימו את הפעולה הרקורסיבית.

שימו ♥: הפעולה מקבלת ארבעה פרמטרים (מערך שמייצג את מגדלי הנוי, מספר דסקיות שצריך להעביר ושני מוטות – מקור ויעד). את המקום במערך של המוט הנוסף ניתן לחשב.

דף מספר 5

לפניכם פיתרון המשימה המופיעה בדף 4 – הפעולה עם ההשלמות:

```
private static void HanoiTowersSolution(Tower[] towers, int nRings,
                                         int fromPole, int toPole)
{
    int extraPole;
    //          כאן צריך לחשב את מספרו של המוט הנוסף
    extraPole = 3 - (fromPole + toPole);
    //          כאן צריך להעביר  $n - 1$  דסקיות מהמוט המקור למוט הנוסף
    HanoiTowersSolution(towers, nRings - 1, fromPole, extraPole);
    towers[fromPole].MoveDisc(towers[toPole]);
    //          כאן צריך להעביר  $n - 1$  דסקיות מהמוט הנוסף למוט היעד
    HanoiTowersSolution(towers, nRings - 1, extraPole, toPole);
}
```

✓ בדקו את הפעולה – האם קיבלתם את הפיתרון? מדוע?

רמז: בעייתיות של הלולאה האינסופית.

דף מספר 6

לפניכם פיתרון הבעיה המופיעה בדף 5.

הפעולה יוצרת לולאה אינסופית. נפתור את הבעיה כך:

כאשר מספר הדסקיות שווה לאפס, אנחנו לא רוצים להעביר אותן. כלומר, רק אם מספר הדסקיות גדול מאפס נרצה להעביר אותן.

```
private static void HanoiTowersSolution(Tower[] towers, int nRings, int fromPole,
                                         int toPole)
{
    int extraPole;
    extraPole = 3 - (fromPole + toPole);
    if (nRings > 0)
    {
        extraPole = 3 - (fromPole + toPole);

        HanoiTowersSolution(towers, nRings - 1, fromPole, extraPole);

        towers[fromPole].MoveDisc(towers[toPole]);

        HanoiTowersSolution(towers, nRings - 1, extraPole, toPole);
    }
}
```

✓ הוסיפו את הוראת תנאי העצירה ובדקו שוב. האם קיבלתם את הפיתרון הנכון?

דף מספר 7

✓ עליכם למדוד את זמני הריצה של הפעולה *HanoiTowersSolution* על אורכי קלט שונים (מספר דסקיות שונה).

- כדי למדוד את זמני הריצה, הריצו את התכנית מספר פעמים על אורכי קלט שונים.
- מדדו את זמני הריצה בפועל בשניות (בסיום המדידה יודפס כמה שניות נמשכה הריצה של הפעולה).
- סיפרו את מספר ההעברות של דסקיות שמתבצעות עבור כל קלט.
- מלאו בטבלה הבאה את מספר הצעדים וזמני הריצה (בשניות), המתבצעים עבור כמויות שונות של דסקיות.

מספר הדסקיות	מספר ההעברות	זמני הריצה בשניות
2		
3		
4		
5		
6		
7		
8		
9		

שימו ♥: המחלקה *Tower* מאפשרת ליצור מוט שבו יאוחסנו לכל היותר 9 דסקיות.

כדי למדוד את משך זמן ריצת הפעולה במחשב, בשניות, עליכם להשתמש במחלקה *Stopwatch* הקיימת בספרייה הסטנדרטית של שפת *C#*. המחלקה *Stopwatch* שייכת למרחב השמות *System.Diagnostics*. כדי להשתמש במחלקה עליכם להכריז על מרחב השמות כך:
using System.Diagnostics;

לפניכם ממשק חלקי של המחלקה *Stopwatch* :

<i>Stopwatch()</i>	הפעולה הבונה של המחלקה שעון-עצר.
<i>void Start()</i>	הפעולה מתחילה את מניית הזמן.
<i>void Stop()</i>	הפעולה עוצרת את מניית הזמן שעבר מההתחלה של מניית הזמן בפעם האחרונה.
<i>long ElapsedMilliseconds</i>	התכונה מחזירה את הזמן שחלף באלפיות השנייה.

הערה : ממשק מלא קיים ונגיש דרך סביבת העבודה.

✓ פרשו ונתחו את תוצאות המדידה. מצאו את מספר ההעברות עבור 10 הדסקיות.

✓ מצאו נוסחה לתיאור מספר העברות עבור n הדסקיות.

דף מספר 8

לפניכם פיתרון המשימה המופיעה בדף 7.

כדי למדוד את משך ריצת הפעולה במחשב, במילי-שניות, הריצו את הקוד הזה :

```
namespace TestHanoiTower
```

```
{  
    class HanoiTower  
    {  
        static void Main(string[] args)  
        {  
            Tower[] towers = new Tower[3];  
            towers[0] = new Tower(5, "0");  
            towers[1] = new Tower("1");  
            towers[2] = new Tower("2");  
            Stopwatch watch = new Stopwatch();  
            watch.Start();  
            HanoiTowersSolution(towers, 5, 0, 2);  
            watch.Stop();  
            Console.WriteLine(watch.ElapsedMilliseconds / 1000.0);  
        }  
    }  
    ...  
}
```

תוצאות המדידה :

מספר הדסקיות	מספר ההעברות	זמני הריצה בשניות
2	3	1.899
3	7	4.566
4	15	9.494
5	31	19.759
6	63	39.879
7	127	80.529
8	255	161.42
9	511	323.621

✓ מספר ההעברות עבור 10 הדסקיות הוא 1023.

✓ מספר ההעברות עבור n הדסקיות הוא $2^n - 1$.