בס"ד

מחלקה גרפית לעץ הבינארי של Unit4

מחלקה גרפית זו מיועדת לשימוש קל ופשוט של התלמידים בתוכניות ConsoleApplication בהם משתמשים במהלך הלימוד של יחידת עיצוב התוכנה (unit4) מבלי לחשוף אותם לדרישות הנגזרות מתוכניות גרפיות.

מחלקה זו הינה מחלקה גנרית כך שהיא תומכת בעצים בינאריים מכל הסוגים הנלמדים בהתאם לתוכנית הלימודים של יחידת העיצוב.

❖ המחלקה תאפשר :

1. **הצגת גרפית של עץ בינארי (התואם ל- unit4).**
2. **עידכון ושינוי של עץ המוצג בצורה גרפית.**
3. **החזרת עץ בינארי (התואם ל- unit4) מתוך החלון הגרפי.**

❖ במהלך ההצגה הגרפית ניתן:

1. **לשנות ולעדכן את העץ בצורה נוחה וגרפית, תוך שימוש בעכבר.**
   ניתן להוסיף או למחוק צמתים בעץ וכן ניתן לשנות ערך של כל צומת שמוצג.
2. **לטעון ו/או לשמור את העץ הגרפי בקובץ במחשב.**
3. **לשנות את עיצוב ההצגה של העץ הגרפי מבחינת צבעים וגופן.**
   את העיצוב ניתן לשמור בקובץ לשימוש חוזר.

<u>הוראות שימוש טכניות:</u>

המחלקה מצורפת בקובץ dll בשם: VisualTree.dll

כדי להשתמש במחלקה יש לצרפה לפרוייקט ע"י References →Add Referance
(בדיוק כמו שמוסיפים לפרוייקט את unit4.dll)

ולהוסיף בקוד : `using VisualTree;`

למחלקה VisualTree שתי פעולות סטטיות בהן נשתמש:
(כלומר, לא צריך לייצר עצם ממחלקת VisualTree)

1. `VisualBinTree<char>.DrawTree(BinNode<char> bt)` - **פעולה המציגה את המסך גרפי ומאפשרת הצגה / עידכון / שמירה וטעינה של עץ בינארי.**
   אם bt הוא עץ קיים, נתחיל את הפעולה מהצגת העץ שנשלח.
   אם bt הוא null, נקבל במסך הגרפי רק את שורש העץ עם ערך ברירת מחדל (0) .
   אם הפעלה נקראה ללא שליחת פרמטר, נקבל את המסך הגרפי ללא הצגת העץ.

2. `BinNode<int> bt = VisualBinTree<int>.GetTree()`
   פעולה המחזירה לתוכנית עץ בפורמט של unit4 מהמסך הגרפי.

<u>דוגמה</u>

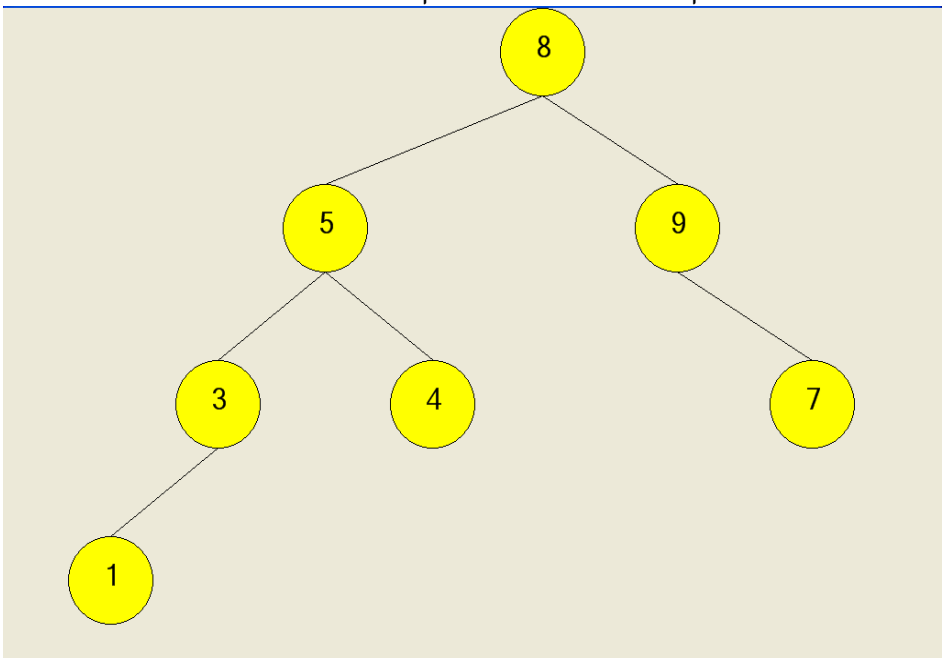
לפניכם קוד הבונה עץ בינארי בשם bt:

```
BinNode<int> bt = new BinNode<int>(new BinNode<int>(1), 3, null);
bt = new BinNode<int>(bt, 5, new BinNode<int>(4));
bt=new BinNode<int>(bt, 8, new BinNode<int>(null, 9, new BinNode<int>(7)));
```

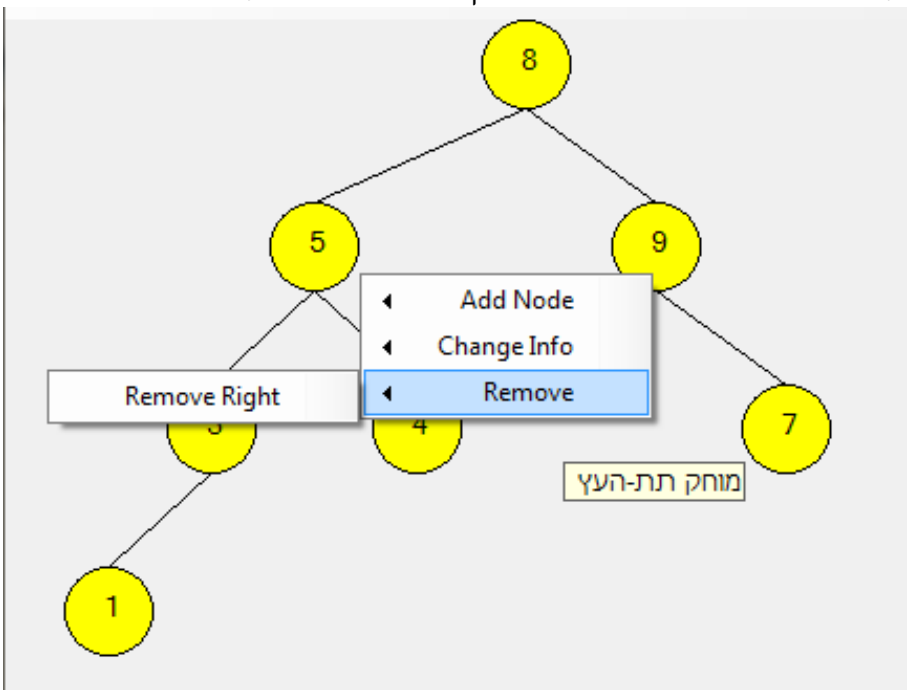
ההוראה שתצייר את העץ היא:

```
BinNode<int> bt = VisualBinTree<int>.DrawTree(bt);
```

הייצוג הגראפי שנקבל באמצעות המחלקה הגראפית VisualTree יהיה:



ע"י בחירת צומת בכפתור ימני נקבל את תפריט העריכה המתאים לצומת :



<u>לדוגמא:</u>
בחירת הצומת עם הערך 9 תוך שימוש בכפתור הימני תפתח תפריט צף המכיל את הפעולות האפשריות על הצומת הנבחר:
הוספת צומת (בן שמאלי בלבד), עידכון ערך הצומת, מחיקת צומת-בן (בן ימני בלבד).

ההוראה שמחזירה את העץ המוצג לתוכנית היא:

```
BinNode<int> bt = VisualBinTree<int>.GetTree();
```

❖ תפריט ראשי, בחלון הגרפי :

<u>פקודות בתפריט ראשי:</u>

- **קובץ** : ניתן לבחור ***קרא*** או ***שמור*** של קובץ בעל סיומת v3.
הקבצים מכילים ייצוג טקסטואלי של עצים בינאריים .
בחירה של אחת מהאפשרויות של התפריט קובץ (קרא / שמור) מקפיץ למשתמש חלון סייר המאפשר "לשוטט" במחשב המקומי ולאתר את הקובץ המדובר או לשמור את הקובץ בתיקיה מסויימת.
תיקיית ברירת המחדל לקריאה / שמירה הינה c:/visualTree אך ניתן לקרא קובץ מכל מקום במחשב.

- **איתחול**: מאתחל את העץ המוצג לצומת אחד, השורש, המכיל את הערך 0.

- **הצג סריקה**: בחירה על התפריט הנ"ל מקפיץ חלון המכיל את 4 אפשרויות הסריקה שנתבקשנו להציג: הצגה בסדר תוכי, הצגה בסדר סופי, הצגה בסדר תחילי, הצגה ע"פ רמות.
התפריט מכיל אפשרויות בחירה, ע"י √, של כל אחת מצורות הסריקה הללו. בעת בחירת אופציה נוספת לסריקה תוצג גם הסריקה הזו בחלון הקופץ ואם הורדה אופציה מסויימת מהבחירה, אז ההצגה שלה יורדת מהחלון הקופץ (אין חובה לסגור ולפתוח את החלון הקופץ מידי שינוי בבחירת האופציות).

- **עיצוב**: ניצן לעצב את הצגת העץ בחלון הגרפי כרצוננו.
ניתן לבחור : צבע צומת / צבע עץ / צבע רקע צבע קו / גופן ו..לשנות אותו כרצוננו.
בכל אחת מהבחירות הללו נקבל חלון קופץ המאפשר לנו לבחור בצורה קלה את הצבע / גופן.
את העיצוב ניתן לשמור בקובץ (שיישמר אף הוא בתיקיית c:/visualTree). בכל הרצה נטען העיצוב מתוך הקובץ שנשמר קודם, אם קיים.

- **סיים**: החלון הגרפי מסתיים וחוזרים לתוכנית שזימנה אותו, ל-ConsoleApplication ולפעולה שכתבנו ואשר זימנה את הצגת העץ הגרפי.

- **אודות**: חלון אודות.

<u>לסיכום:</u>
המחלקה הגראפית ניתנת לשימוש כתוספת לכל תכנית המשתמשת ב-Unit4 ועוסקת בעצים.
הצגת ואפשרויות העידכון הגראפי של העץ עוזרת מאוד בבניית עץ ובמעקב אחר ביצוע תכניות ובעיקר אלו המשנות את העצים.
אפשרויות טעינה ושמירה של עצים תוך שימוש בקבצים מאפשר למורה לספק לתלמידים עצים מוכנים המהווים בסיס הנדרש מהתלמיד לשם פתרון שאלת העצים, ע"פ תוכנית הלימודים.
אספקת עצים מוכנים בצורה זו חוסכת לתלמיד זמן רב שעד עכשיו "בוזבז" ע"י התלמיד לשם בניית העץ (שלא תמיד היה יוצא כמו שהתלמיד התכוון אליו) וכך התלמיד יכול להתרכז בפתרון הבעיה שנשאל על עץ תקין ומתאים לבעיה.
חסכון זה מאפשר לתלמיד לתרגל יותר שאלות עצים בזמן נתון.

<u>דוגמא לשימוש:</u>

בחרתי להביא כדוגמה משימה המשלבת בין מיקום בעץ לערכי כל צומת .

המשימה היא אינטרפרטציה לשאלה מהספר "עצים בינאריים" של איתן ראט וחיים אברבוך.

ניתן ליצג עץ בינארי על ידי קבוצה של מספרים (מחרוזות המכילות 1 ו-2) בדרך הבאה:

ניתן ליצג כל צומת בעץ על ידי מספר.

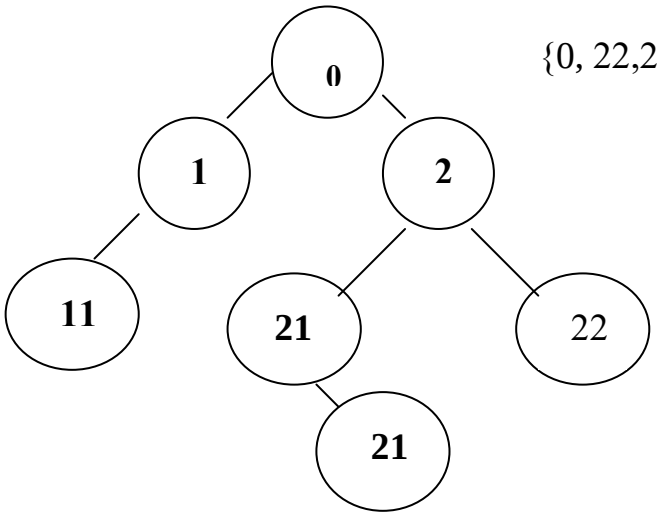
שורש העץ מיוצג ע"י 0 .

את הבן השמאלי של השורש נייצג על ידי המספר 1 , ואת הבן הימני של השורש נייצג על ידי המספר 2.

אם המספר X מיצג צומת מסויים בעץ אז הבן השמאלי שלו מיוצג על ידי המספר X1

ואת הבן הימני נייצג על ידי המספר X2.

דוגמא: לפניך עץ המיוצג על ידי המחרוזות {0, 22,212 ,1,2,11,21}



<u>משימה:</u>

כתוב פעולה המקבלת עץ בינארי T **ומעדכן** את ערכו של כל צומת כך שיכיל את המספר המייצג את אותה צומת:

רצ"ב הקוד עם הסבר (התכנית תצורף למייל גם כתכנית).

בשלב ראשון נבנה עץ עם ערכים כלשהם ונציג אותו בצורה גראפית, כדי לראות ולוודא שאכן קיבלנו את העץ הבינארי שהתכוונו אליו.

בשלב שני כל צומת מקבל את המספר הנדרש בשאלה לפי מיקומו והעץ מוצג בשנית

```
using System;
using System.Text;
using Unit4.CollectionsLib;  // unit4
using VisualTree;            // המחלקה הגראפית להצגת עץ בינארי

namespace Tree2011
{
 class Program
 {
   static void Main(string[] args)
   {
     // בניית עץ בינארי, ע"פ הדרך המסורתית
    BinNode<int> bt1 = new BinNode<int>(9);
    BinNode<int> bt2 = new BinNode<int>(bt1,6,null);
    BinNode<int> bt3 = new BinNode<int>(3);
     BinNode<int> bt4 = new BinNode<int>(null,5,bt3);
     BinNode<int> bt5 = new BinNode<int>(2);
     BinNode<int> bt6 = new BinNode<int>(bt4,8,bt5);
     BinNode<int> bt = new BinNode<int>(bt2,0,bt6);

      VisualBinTree<int>.DrawTree(bt); // הצגת העץ, בצורה גראפית

      Change(bt);  // זימון לפעולה שכתבנו ואשר מעדכנת את העץ

      VisualBinTree<int>.DrawTree(bt); // הצגת העץ המעודכן, בצורה גראפית
   }

    public static void Change(BinNode<int> b)
    {
        if (b != null)
        {
            if (b.GetInfo() == 0)
            {
                if (b.GetLeft() != null)
                    b.GetLeft().SetInfo(1);
                if (b.GetRight() != null)
                    b.GetRight().SetInfo(2);
            }
            if (b.GetLeft() != null)
                b.GetLeft().SetInfo(b.GetInfo() * 10 + 1);
            if (b.GetRight() != null)
                  b.GetRight().SetInfo(b.GetInfo() * 10 + 2);
            Change(b.GetLeft());
            Change(b.GetRight());
        }
     }

  }   // end class
}   // end nameSpace
```

ניתן להחליף את השלב הראשון של בניית העץ מהצורה המסורתית לצורה הגראפית או לטעינה של צץ קיים מתוך קובץ נתון ו..אז כל שנותר זה לזמן את הפעולה שכתבנו עבור השאלה (Change) ולהציג שוב את העץ שעודכן, כדי לוודא שאכן קיבלנו את התוצאה הרצוייה.

הפעם הקוד יהיה:

```
using System;
using System.Text;
using Unit4.CollectionsLib;  // unit4
using VisualTree;            // המחלקה הגראפית להצגת עץ בינארי

namespace Tree2011
{
 class Program
 {
   static void Main(string[] args)
   {
                  //  הצגת עץ מינימלי, בצורה גראפית, לשם בנייתו
    BinNode<int> bt = VisualBinTree<int>.DrawTree(null);
                  //   או ...רק הצגת החלון הגראפי לשם טעינת העץ מקובץ
//  BinNode<int> bt = VisualBinTree<int>.DrawTree();

    Change(bt);  //זימון לפעולה שכתבנו
    VisualBinTree<int>.DrawTree(bt);  // הצגת העץ המעודכן, בצורה גראפית
   }
}
```