פרק 10
עץ בינרי

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

חלק מהמצגת מבוסס על תכנים מהספר:

עצים בינאריים

של:

איתן ראט וחיים אברבוך

הוצאת למידה בהנאה, 2002

אנו מודים להם על האישור להשתמש בחומרים

מה בשיעור

1. עץ כללי
2. מושגים
3. עץ בינרי
4. חוליה בינרית
5. עץ חוליות בינרי
6. עץ חוליות בינרי כמבנה רקורסיבי
7. פעולות על עצי חוליות בינריים
8. עץ-חיפוש-בינרי

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ

אנו רוצים לשמור מידע על בני משפחה כולל קשרי המשפחה:

ננסה לשמור את בני משפחת אברהם אבינו במערך:

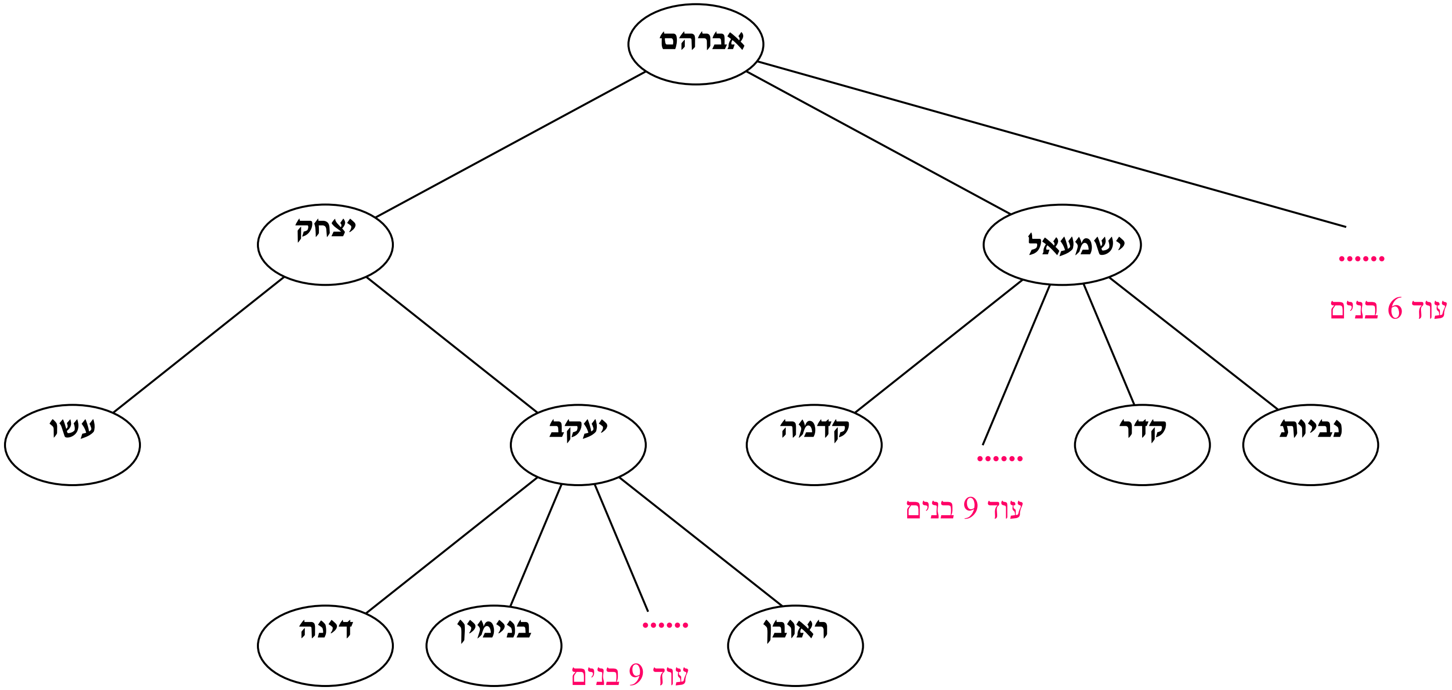
אברהם	יצחק	ישמעאל	עשו	יעקב	קדמה	קדר	נביות

אומנם יש ייצוג לבני המשפחה, אך לא ניתן לדעת מה הם קשרי המשפחה.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

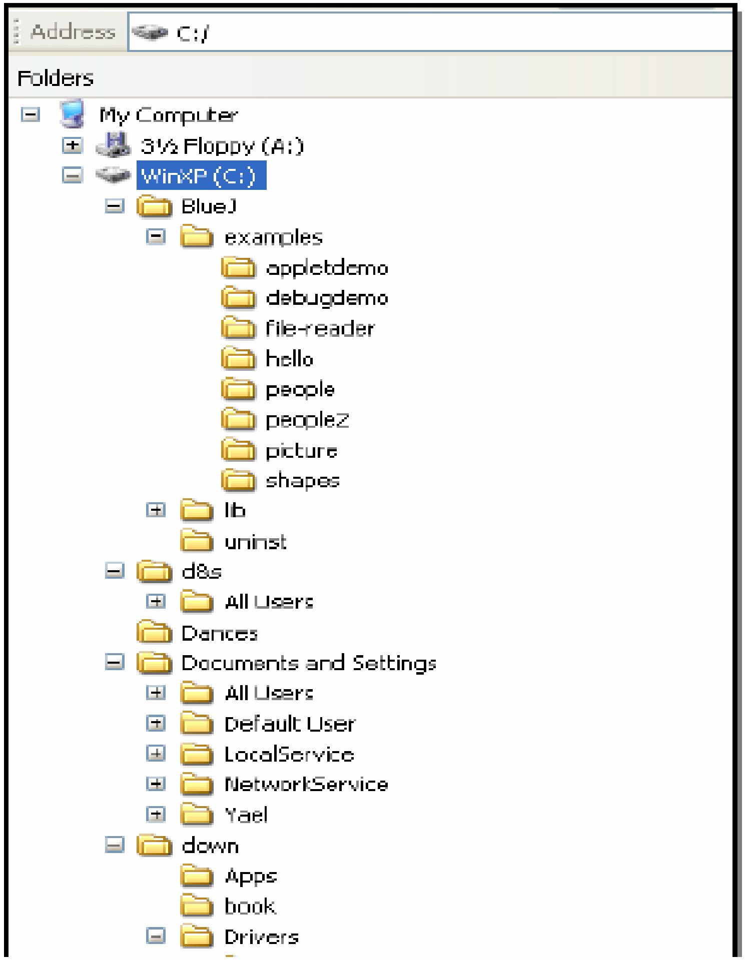
דוגמה לעץ-משפחה

ניתן לתאר את קשרי המשפחה באמצעות אילן יוחסין :



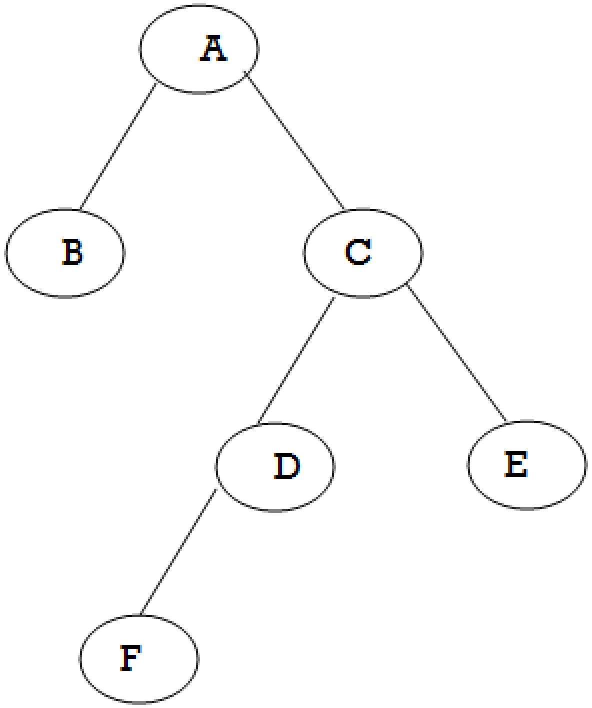
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמה נוספת לעץ - מערכת קבצים



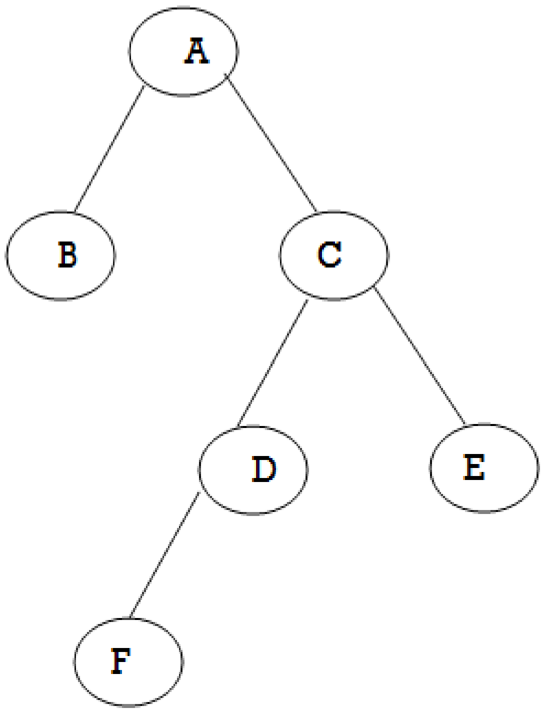
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עצים: מושגים



- **שורש העץ**-הצומת העליון בעץ הצומת A
- **צומת**-כל איבר בעץ הוא צומת
- **בן ימני ובן שמאלי**-הצמתים B הוא שמאלי של A
- **הורה**-לדוגמה D הוא אבא של F
- **אחים**-הם שני צמתים ששניהם בנים לאותו אבא
- **צאצא**-לדוגמה F הוא צאצא של C, וכל איברי העץ
- הם צאצאים של שורש העץ A
- **הורה קדמון**-C הוא הורה-קדמון של F
- אך גם של D ו- E שורש העץ A הוא אב-קדמון של כל שאר הצמתים
- **עלה**-הוא צומת ששני תת-עציו הם עצים ריקים לדוגמה הצמתים E ו-F
- **תת-עץ-ימני של צומת**-הצמתים C,D,E,F מהווים תת-עץ ימני של A
- **תת-עץ-שמאלי של צומת**-הצמתים D ,F מהווים את תת-העץ השמאלי של C

עצים: מושגים המשך



- **רמה של צומת-** היא מספר הקטעים במסלול בין

שורש העץ לצומת הרמה של השורש היא 0 רמתו של F 3

- **רמה בעץ-** היא קבוצת כל הצמתים בעץ,שרמתם שווה
- למשל רמה 2 מכילה הצמתים D,E
- **רמה מלאה-** רמה בעץ שקיימים בה כל הצמתים
- למשל רמה 1 היא מלאה
- **עץ-שלם**-עץ שכל רמותיו מלאות
- עץ מלא-עץ שאין בו בנים יחידיים

מגבלות המגדירות מבנה של עץ

- קיים צומת אחד בדיוק ללא הורה ; צומת זה קרוי **שורש** העץ.
- לכל צומת שאינו השורש יש הורה יחיד.
- כל צומת הוא צאצא של השורש.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

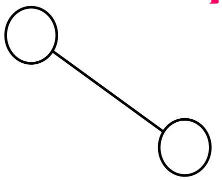
עץ בינרי (Binary Tree)

- עץ שבו לכל צומת יש לכל היותר שני ילדים נקרא **עץ בינרי** (Binary Tree)
- הילדים נקראים **ילד שמאלי וילד ימני**.
- הילד השמאלי הוא שורש של **תת עץ שמאלי**.
- הילד הימני הוא שורש של **תת עץ ימני**.

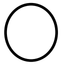
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות לעצים בינריים

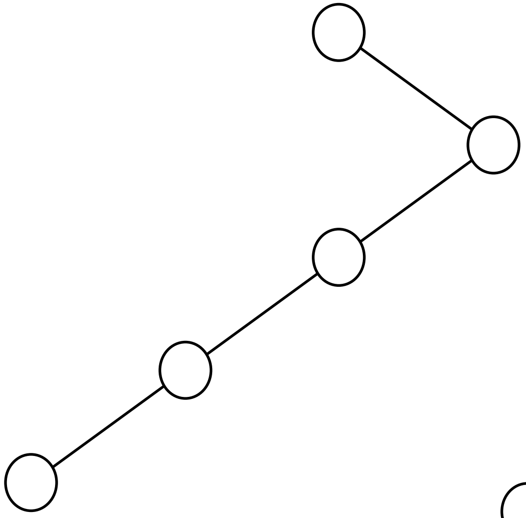
א.



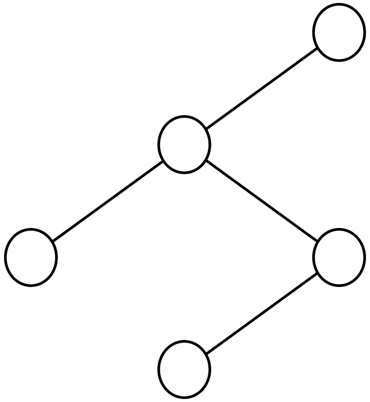
ב.



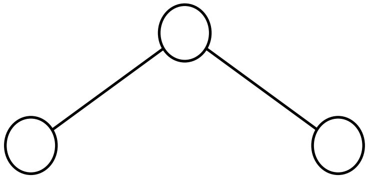
ג.



ד.

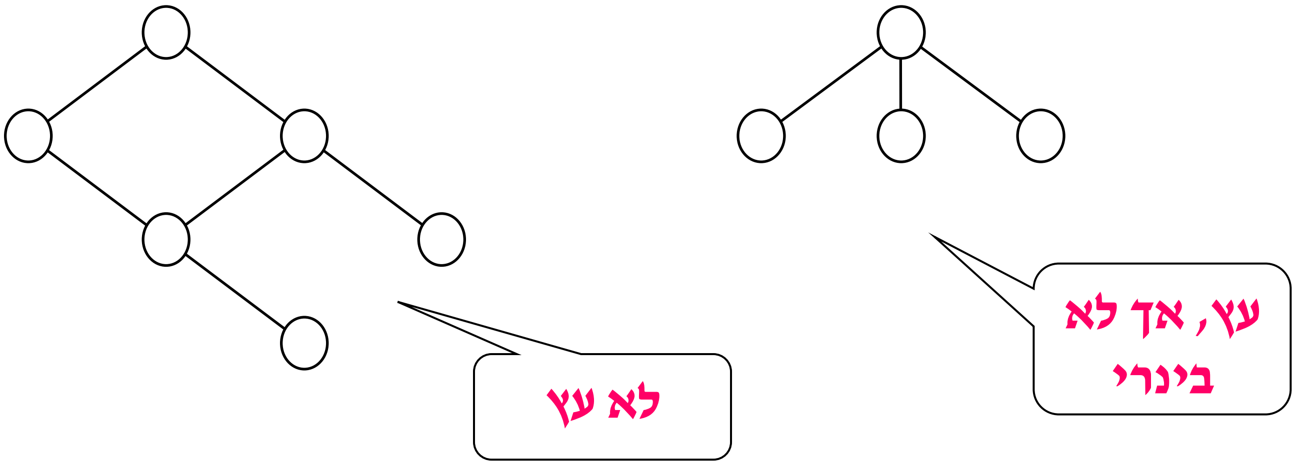


ה.



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות לעצמים שאינם עצים בינריים



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

חוליה בינרית BinTreeNode<T>

לחוליה זו 3 תכונות:

- info : הערך
- left : הילד השמאלי
- right : הילד הימני

לכל אחת מהתכונות קיימות פעולות Set ו-Get.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

ממשק החוליה בינרית

BinTreeNode (T x)	הפעולה בונה חוליה בינרית. ערך החוליה הוא x וערך שתי ההפניות שלה הוא **null**
BinTreeNode (BinTreeNode<T> left, T x, BinTreeNode<T> right)	הפעולה בונה חוליה בינרית שערכה יהיה x. left ו-right הן (הפניות אל) הילד השמאלי והימני שלה. ערכי ההפניות יכולים להיות **null**
T GetInfo()	הפעולה מחזירה את הערך של החוליה
void SetInfo (T x)	הפעולה משנה את הערך השמור בחוליה ל-x
BinTreeNode<T> GetLeft()	הפעולה מחזירה את הילד השמאלי של החוליה. אם אין ילד שמאלי הפעולה מחזירה **null**
BinTreeNode<T> GetRight()	הפעולה מחזירה את הילד הימני של החוליה. אם אין ילד ימני הפעולה מחזירה **null**
void SetLeft (BinTreeNode<T> left)	הפעולה מחליפה את הילד השמאלי בחוליה left
void SetRight (BinTreeNode<T> right)	הפעולה מחליפה את הילד הימני בחוליה right
string ToString()	הפעולה מחזירה מחרוזת המתארת את הערך השמור בחוליה

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

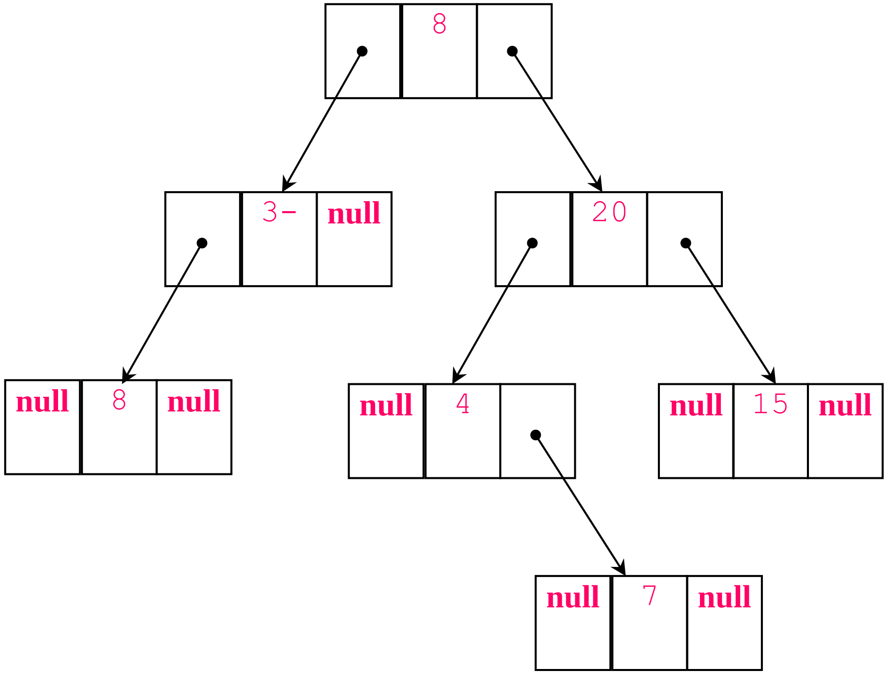
שאלה

BinTreeNode<T> ממשו את המחלקה ?

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ חוליות בינרי

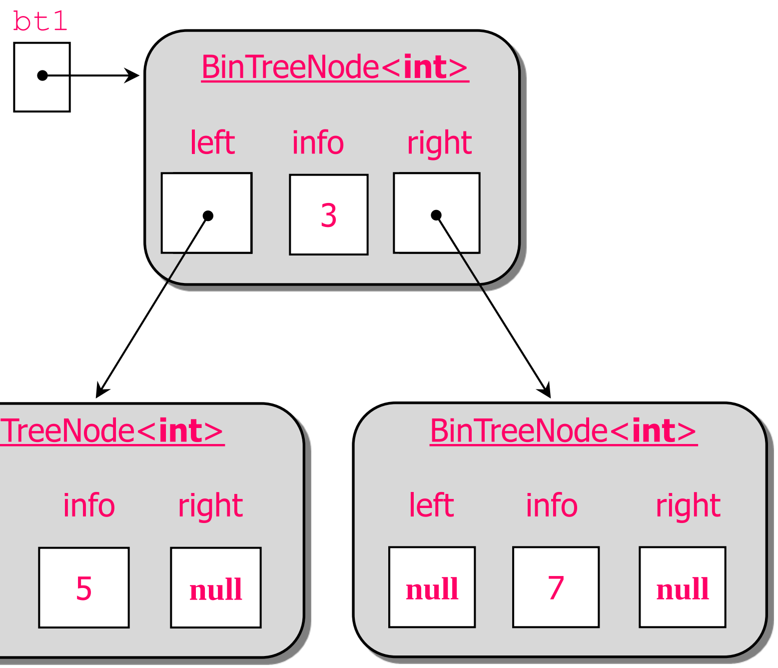
באמצעות החוליה הבינרית ניתן לבנות עץ חוליות בינרי :



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בניית עץ חוליות בינרי

```
BinTreeNode<int> bt1 = new BinTreeNode<int>(3);
bt1.SetLeft (new BinTreeNode<int>(5));
bt1.SetRight (new BinTreeNode<int>(7));
```



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

שאלה

עץ עלה הוא העץ הקטן ביותר שיכול להתקיים.

פעמים רבות ניעזר בבדיקה האם עץ מסוים הוא עץ עלה.

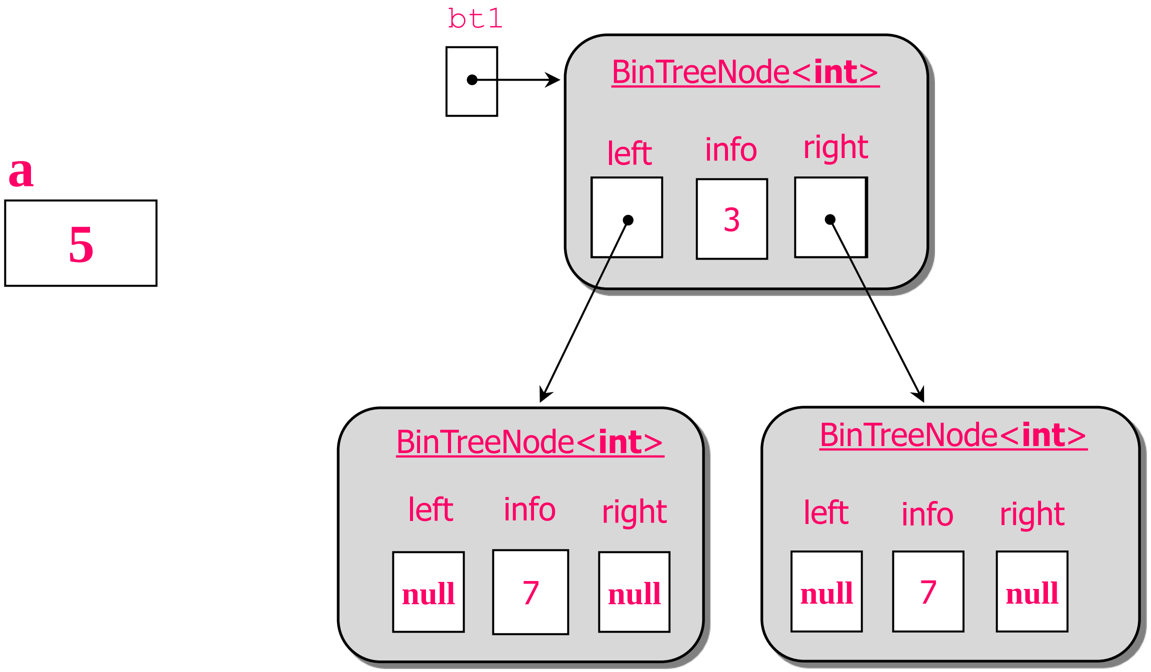
? כתבו קטע קוד הבודק האם חוליה בינרית נתונה היא עץ עלה.

```
public static bool IsLeaf(BinTreeNode<int> bt)
{
   return (bt.GetLeft()==null)&&(bt.GetRight()==null);
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

תנועה על עץ חוליות בינרי ושינויו

```
int a = bt1.GetLeft().GetInfo();
bt1.GetLeft().SetInfo(7);
```



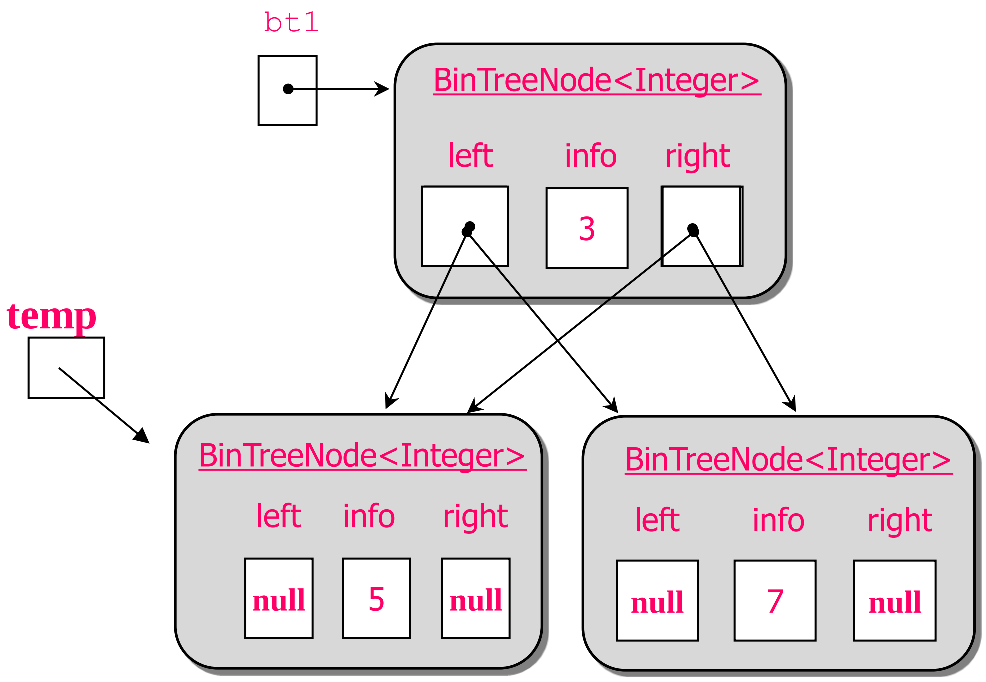
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

תנועה על עץ חוליות בינרי ושינויו

BinTreeNode<**int**> temp = bt1.GetLeft();

bt1.SetLeft (bt1.GetRight());

bt1.SetRight (temp);



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

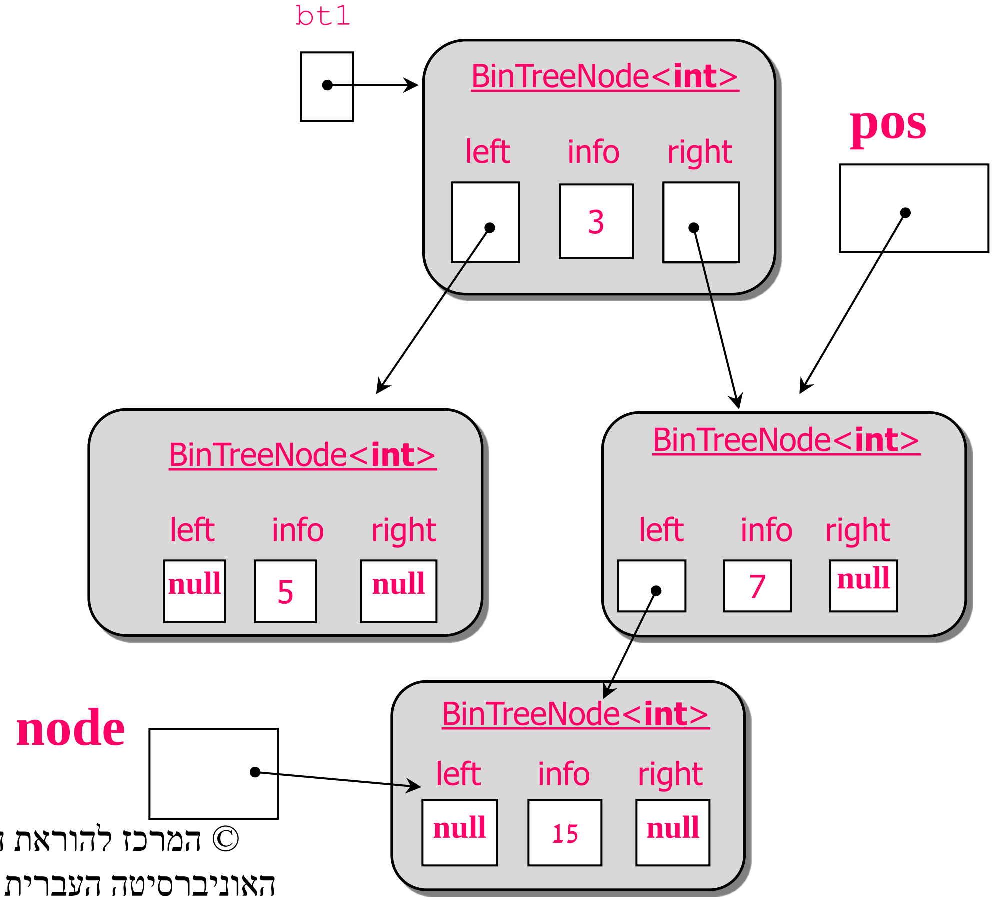
הכנסה

- הכנסה בקצה העץ:

BinTreeNode<**int**> node = **new** BinTreeNode<**int**>(15);

BinTreeNode<**int**> pos = bt1.GetRight();

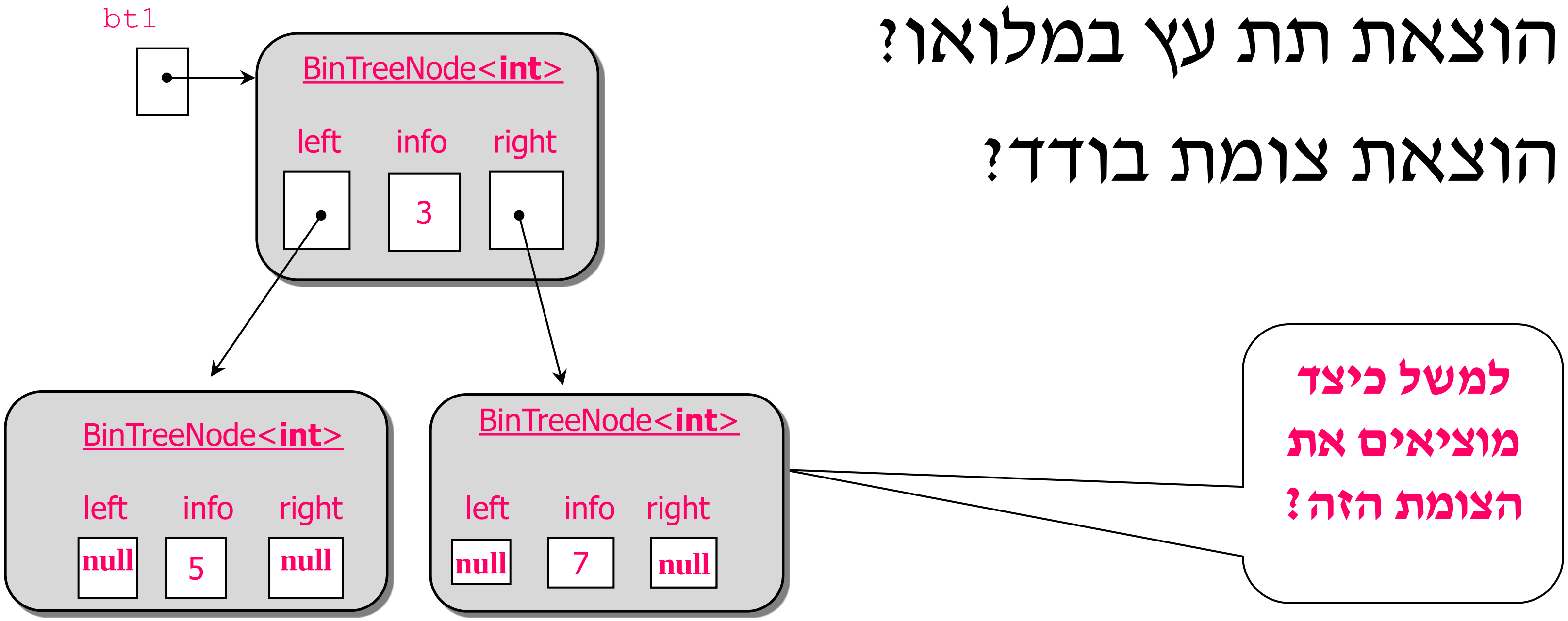
pos.SetLeft (node);



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

הוצאה

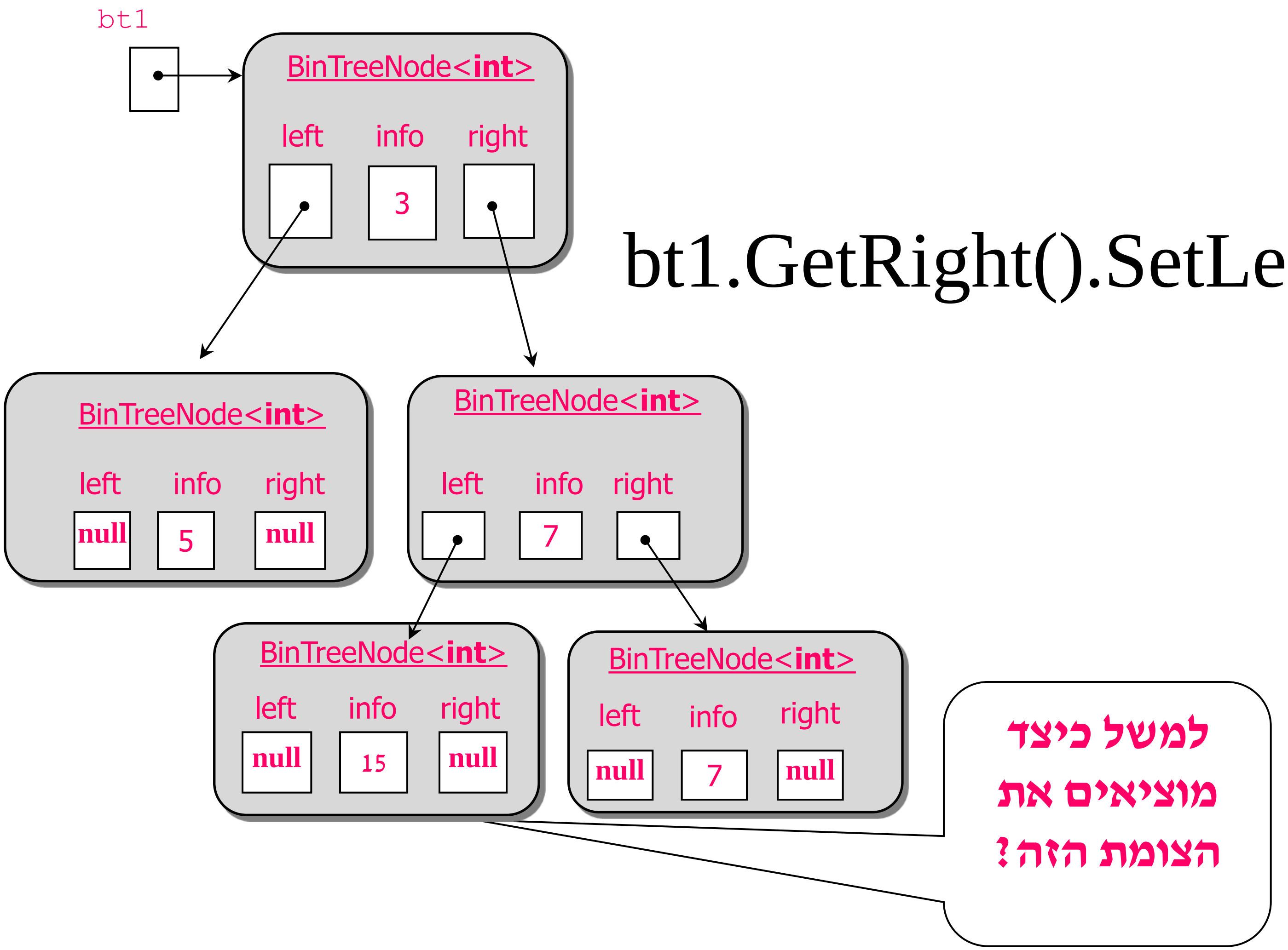
- הוצאת עץ עלה?
- הוצאת תת עץ במלואו?
- הוצאת צומת בודד?



bt1.SetRight(null);

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

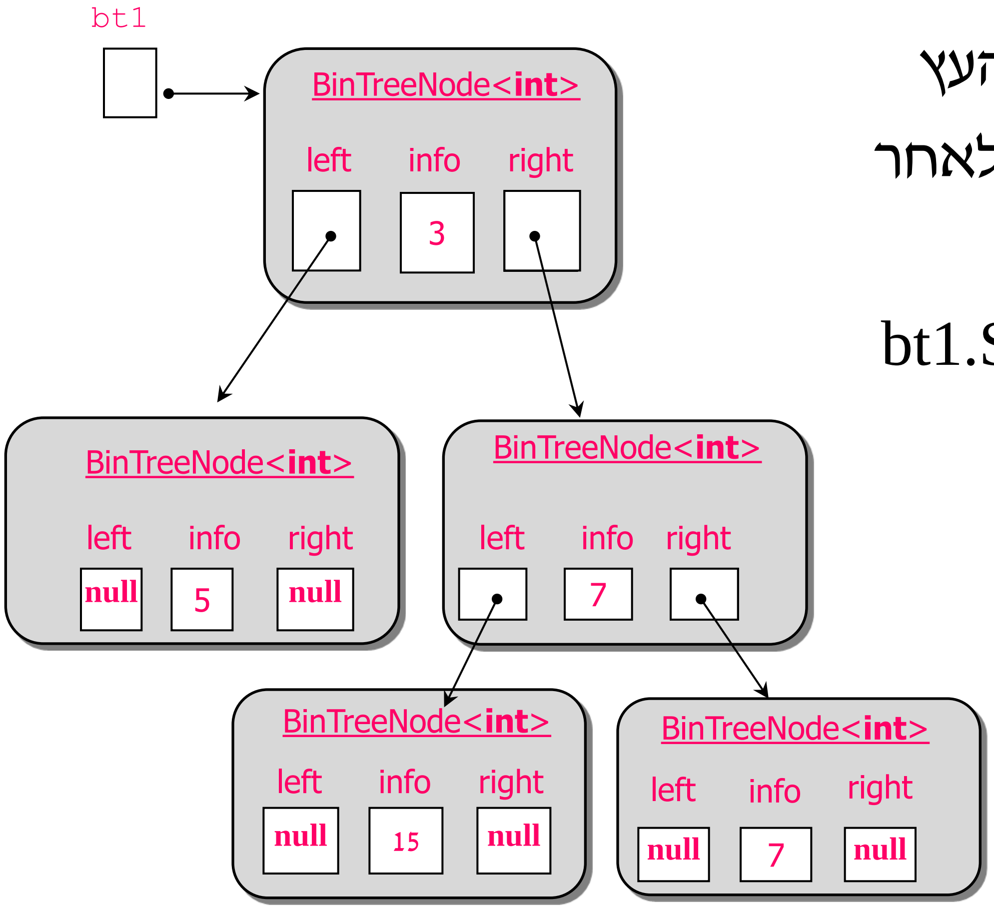
הוצאה



bt1.GetRight().SetLeft(null);

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

אי שמירה על מבנה העץ



קל מאוד להרוס את מבנה העץ

ציירו את המבנה המתקבל לאחר

ביצוע השורה:

bt1.SetLeft (bt1.GetRight());

ולאחר ביצוע השורה:

bt1.SetLeft (bt1);

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ חוליות בינרי – מבנה רקורסיבי

עץ חוליות בינרי הוא:

- חוליה בינרית יחידה

או

- חוליה בינרית שבה לכל היותר שתי הפניות לעצי חוליות בינריים הזרים זה לזה (אין להם חוליות משותפות)

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סריקות עומק של עץ בינרי

קיימות 3 סריקות עומק של עץ:

- **סריקה בסדר תחילי (preorder traversal).**
- **סריקה בסדר תוֹכי (inorder traversal).**
- **סריקה בסדר סופי (postorder traversal).**

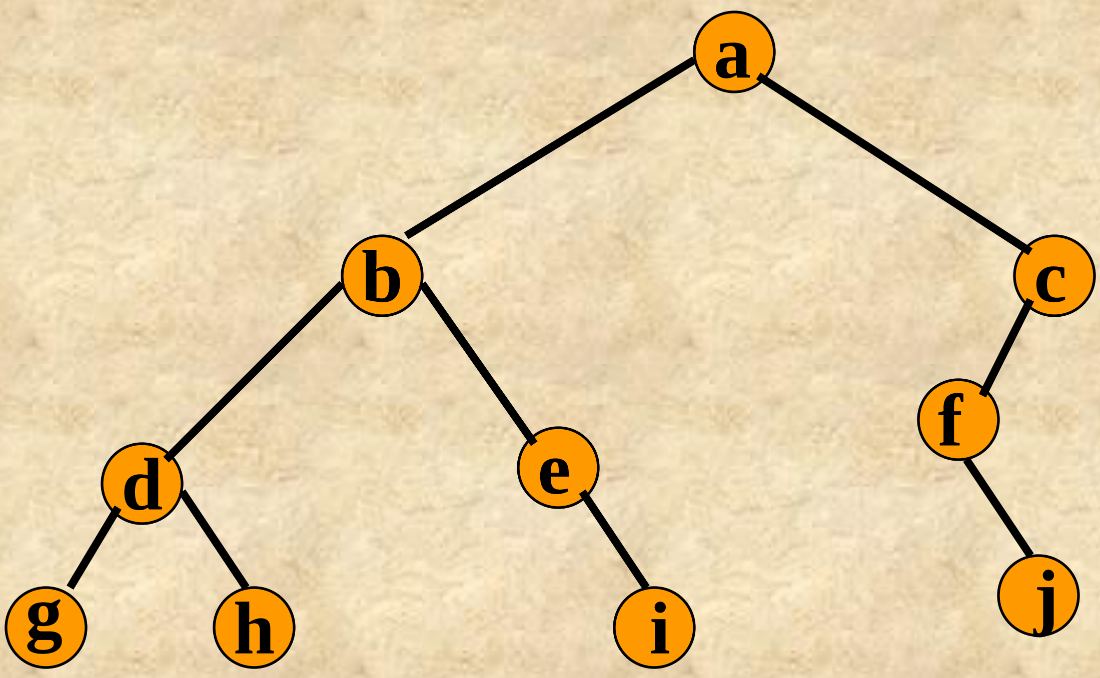
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סריקה תחילית (preorder)

```
//פעולה המדפיסה את העץ לפי סריקה תחילית
public static void PreOrder(BinTreeNode<int> t)
{
    if (t != null)
    {
        Console.WriteLine(t.GetInfo());
        PrintTree(t.GetLeft());
        PrintTree(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

Preorder Example

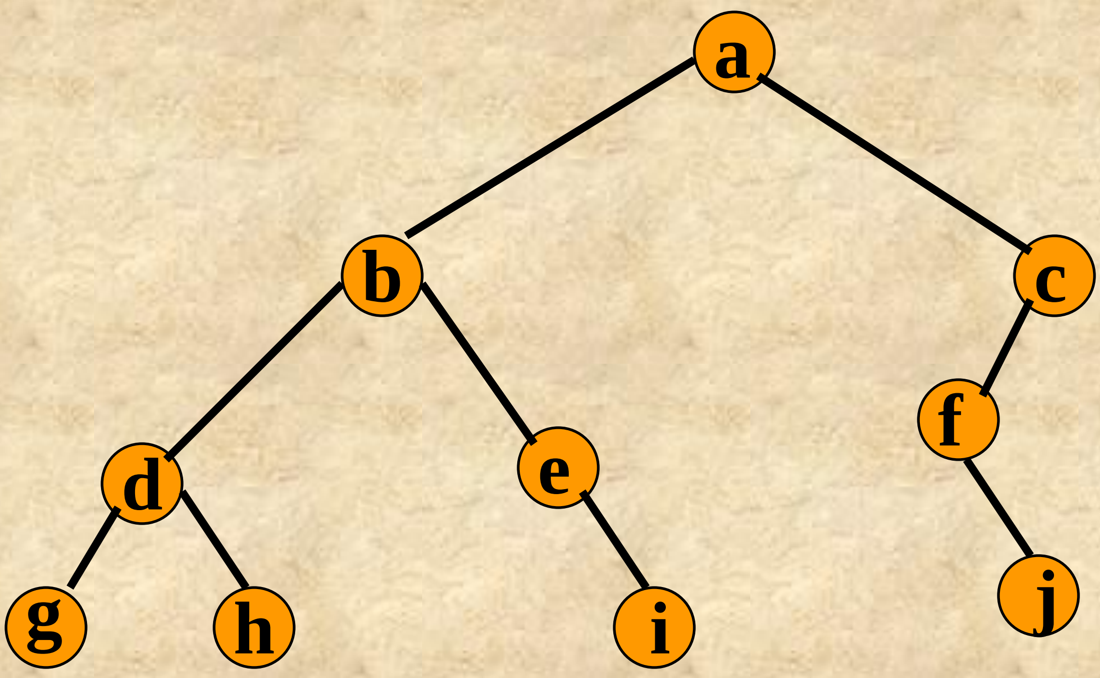


a b d g h e i c f j

סריקה תוכית (inorder)

```
//פעולה המדפיסה את העץ לפי סריקה תוכי
public static void InOrder(BinTreeNode<int> t)
{
    if (t != null)
    {
        PrintTree(t.GetLeft());
        Console.WriteLine(t.GetInfo());
        PrintTree(t.GetRight());
    }
}
```

Inorder Example

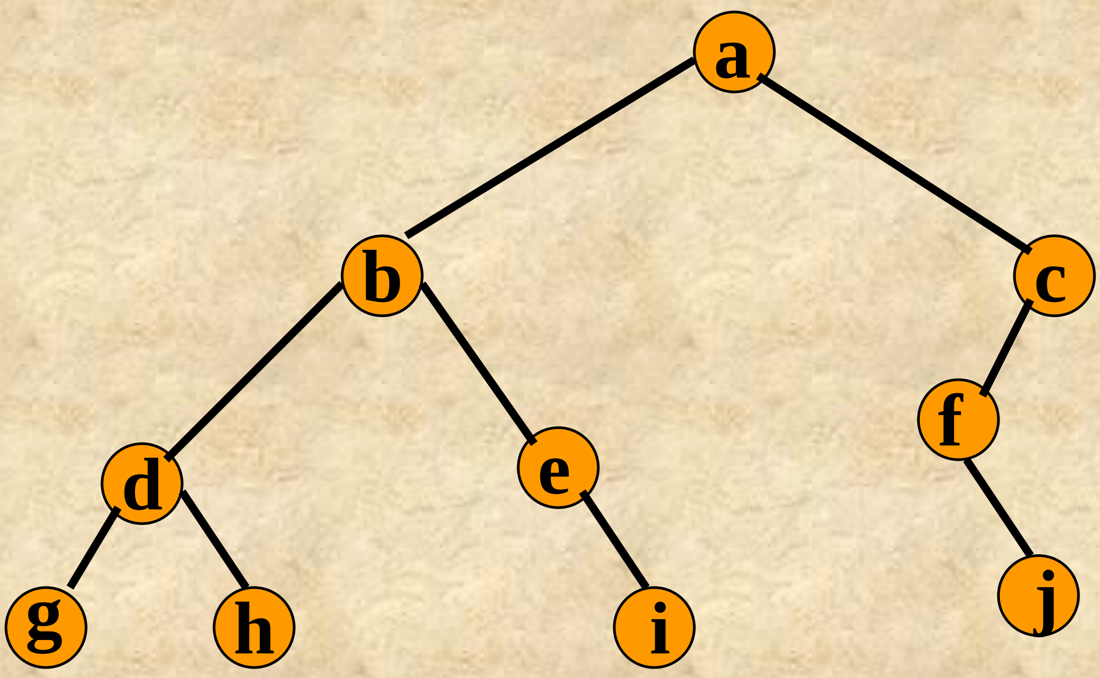


g d h b e i a f j c

סריקה תוכית (postorder)

```
//פעולה המדפיסה את העץ לפי סריקה סופי
public static void PostOrder(BinTreeNode<int> t)
{
    if (t != null)
    {
        PrintTree(t.GetLeft());
        PrintTree(t.GetRight());
        Console.WriteLine(t.GetInfo());
    }
}
```

Postorder Example



g h d i e b j f c a

פעולה הבונה עץ

```
//פעולה בונה עץ
public static BinTreeNode<int> CreateTree()
{
          Console.WriteLine("Enter Node");
          int x = int.Parse(Console.ReadLine());
          int d;
         Console.WriteLine("Enter 0 To Leaf, Enter 1 To Left " + x + " Enter 2 To Right " + x + " Enter 3 Two
Child");
         d = int.Parse(Console.ReadLine());
             if (d == 0) // עלה
                return new BinTreeNode<int>(x);
             else if (d == 1)//בן שמאלי יחיד
               return new BinTreeNode<int>(CreateTree(), x, null);
                else if (d == 2) //בן יחיד ימני
                     return new BinTreeNode<int>(null, x, CreateTree());
  // שני בנים
          else return new BinTreeNode<int>(CreateTree(), x, CreateTree());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

פלט הפעולה

Enter Node

8

Enter **0 To Leaf** ,Enter **1 To Left** 8 Enter **2 To Right** 8 Enter **3 Two Child**

3

Enter Node

7

Enter **0 To Leaf** ,Enter **1 To Left** 7 Enter **2 To Right** 7 Enter **3 Two Child**

1

Enter Node

4

Enter **0 To Leaf** ,Enter **1 To Left** 4 Enter **2 To Right** 4 Enter **3 Two Child**

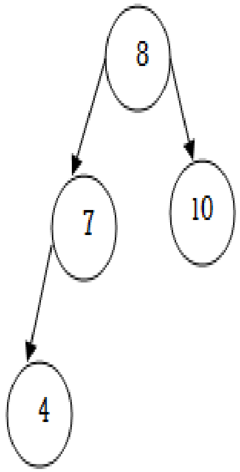
0

Enter Node

10

Enter **0 To Leaf** ,Enter **1 To Left** 10 Enter **2 To Right** 10 Enter **3 Two Child**

0



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

יעילות הסריקות

- בשלושת האלגוריתמים הסריקה מבצעת ביקור יחיד בכל צומת של העץ.
- לכן זמן הריצה של כל אחת מהסריקות הוא לינארי בגודל העץ (שהוא מספר צמתיו).
- סדר הגודל של היעילות הוא O(n).

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות לסריקה

```
//פעולה המדפיסה את העץ לפי סריקה תחילי
public static void PrintTree(BinTreeNode<int> t)
{
    if (t != null)
    {
        Console.WriteLine(t.GetInfo());
        PrintTree(t.GetLeft());
        PrintTree(t.GetRight());
    }
}
```

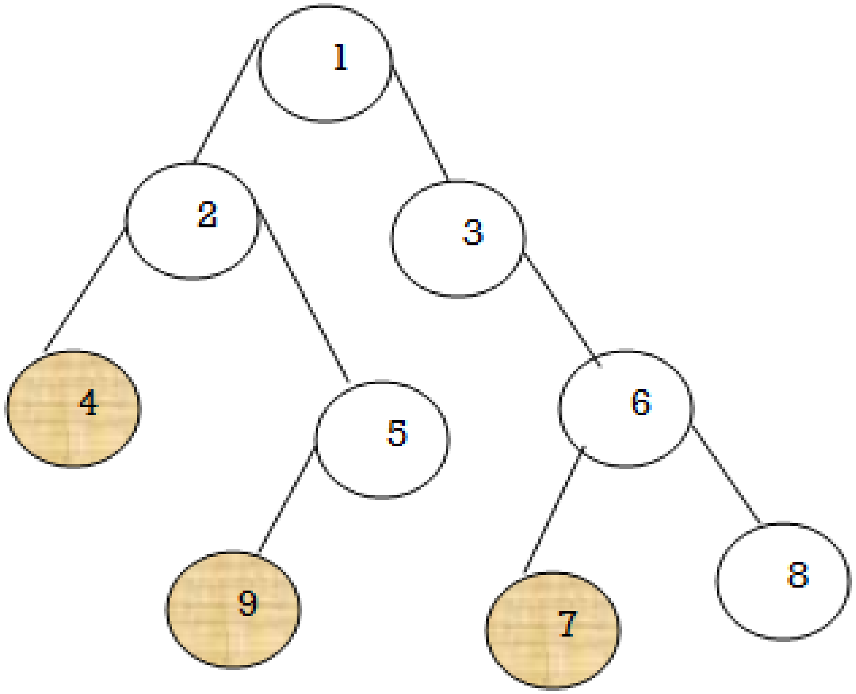
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות לסריקה המשך דוגמה 2

```
//פעולה המדפיסה את כל הצמתים הזוגיים בעץ
public static void PrintEven(BinTreeNode<int> t)
{
    if (t != null)
    {
        if (t.GetInfo()%2==0)
         Console.WriteLine(t.GetInfo());
        PrintEven(t.GetLeft());
        PrintEven(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

הדפסת כל העלים השמאליים



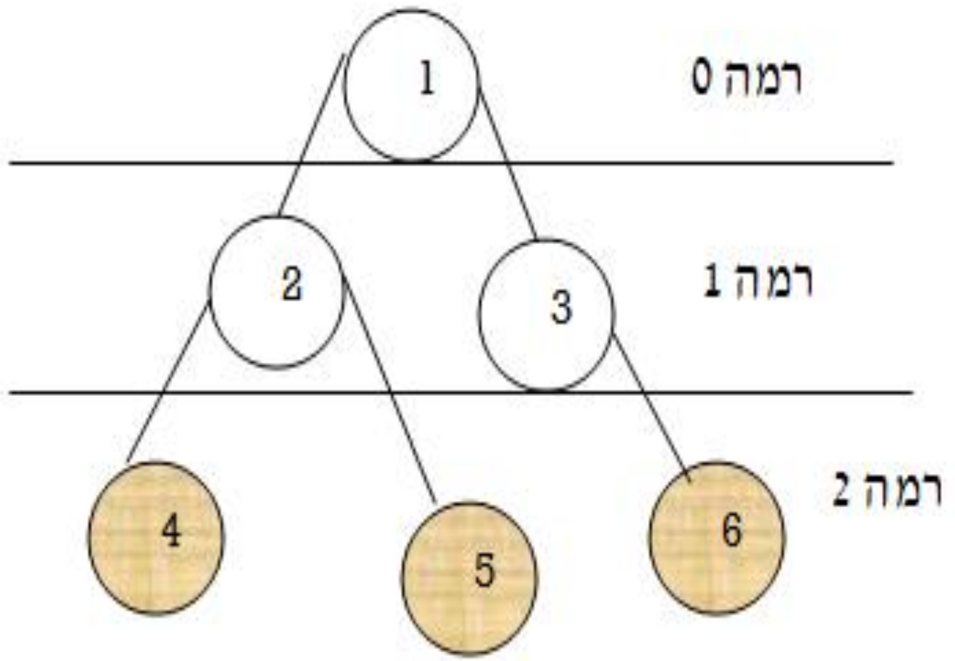
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות לסריקה –פתרון 3

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר//
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה המדפיסה את כל העלים השמאליים//
public static void PrintChildLeftIsLeaf(BinTreeNode<int> t)
{
    if (t != null)
    {
        if (t.GetLeft() != null && IsLeaf(t.GetLeft()))//בן שמאלי וגם עלה
            Console.WriteLine(t.GetLeft().GetInfo());
        PrintChildLeftIsLeaf(t.GetLeft());
        PrintChildLeftIsLeaf(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

הדפסת הצמתים ברמה מסויימת n



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

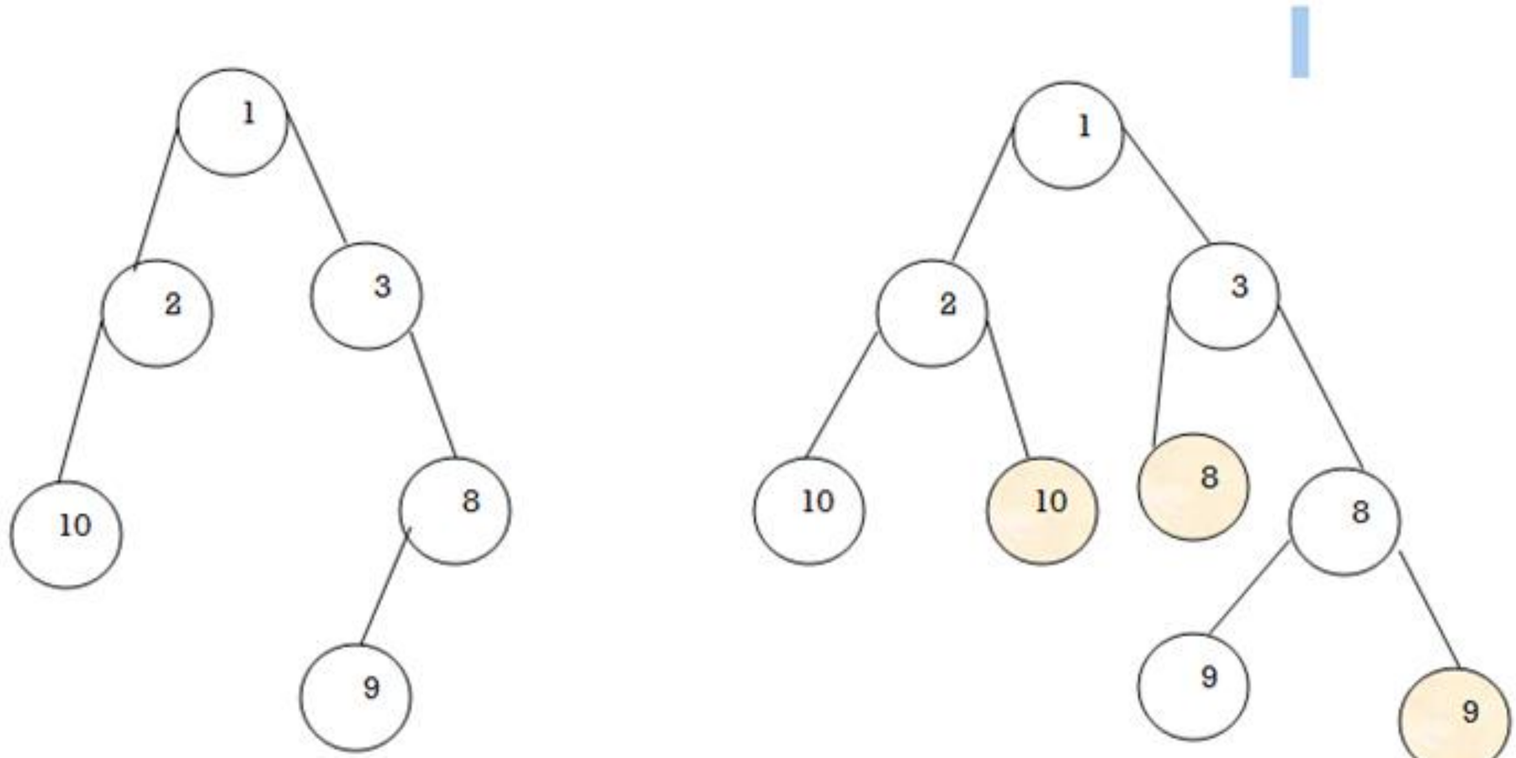
דוגמאות לסריקה – פתרון 4

```
//פעולה מדפיסה את הצמתים ברמה n
public static void PrintLevel(BinTreeNode<int> t, int n)
{
    if (t!=null)
    {
        if (n == 0)
            Console.WriteLine(t.GetInfo());
        PrintLevel(t.GetLeft(), n - 1);
        PrintLevel(t.GetRight(), n - 1);
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמה פתורה –הכנסה,להכניס לכל בן יחיד אח זהה לו

עץ לאחר השינוי



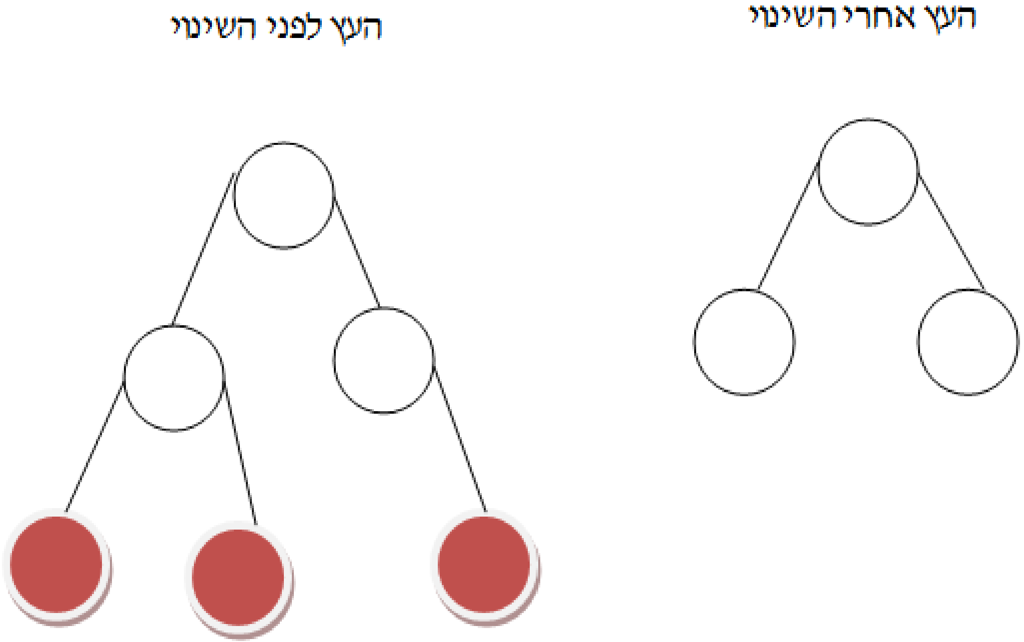
האוניברסיטה העברית בירושלים

פתרון

```
//פעולה מקבלת עץ ומוסיפה לכל בן יחיד אח  הערך שלו יהיה כמו הבן היחיד//
//אם לצומת עלה לא יתבצע דבר//
public static void InsertBrother(BinTreeNode<int> t)
{
    if (t!=null)
    {
        if (t.GetLeft()==null&& t.GetRight()!=null)//בן יחיד ימני//
            t.SetLeft(new BinTreeNode<int>(t.GetRight().GetInfo()));
        if (t.GetRight()==null&& t.GetLeft()!=null)//בן יחיד שמאלי//
            t.SetRight(new BinTreeNode<int>(t.GetLeft().GetInfo()));
        InsertBrother(t.GetLeft());
        InsertBrother(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

מחיקת כל העלים בעץ

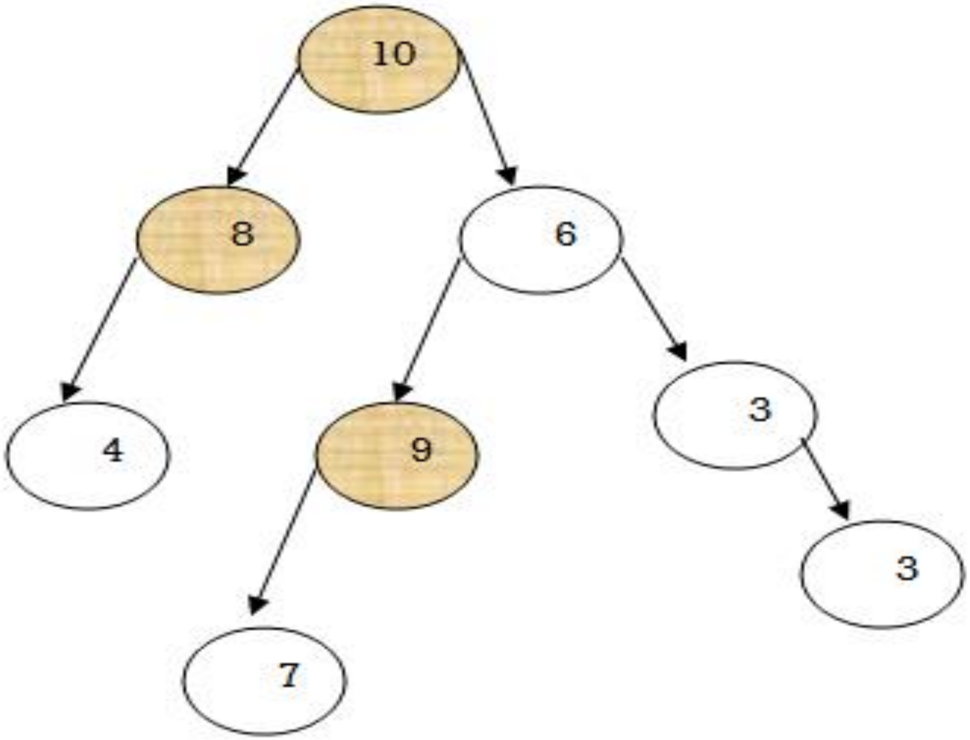


פעולה המחוקת את כל העלים בעץ

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
// ,פעולה המוחקת את כל העלים בעץ
public static void  RemoveAllLeaf(BinTreeNode<int> t)
{
    if (t != null)
    {
        if (t.GetLeft() != null && IsLeaf(t.GetLeft()))
        {
            t.SetLeft(null);
        }
        if (t.GetRight() != null && IsLeaf(t.GetRight()))
        {
            t.SetRight(null);
        }
        RemoveAllLeaf(t.GetLeft());
        RemoveAllLeaf(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

הדפסת כל ההורים הגדולים מהבנים



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

פתרון הדפסת הצמתים הגדולים מהבנים

```
//פעולה מקבלת עץ ומדפיסה את את הצמתים שגדולים מהבנים שלהם
public static void PrintPernt(BinTreeNode<int> t)
{
    if (t != null)
    {
        //קיים שני בנים וההורה גדול מהבנים
        if (t.GetLeft() != null && t.GetLeft().GetInfo() < t.GetInfo()
            && t.GetRight() != null && t.GetRight().GetInfo() < t.GetInfo())
            Console.WriteLine(t.GetInfo());
        //קיים בין יחיד שמאלי והוא קטן מהורה
        if (t.GetLeft()!=null && t.GetRight()==null&&
            t.GetInfo() > t.GetLeft().GetInfo())
            Console.WriteLine(t.GetInfo());
        //קיים בין יחיד ימני והוא קטן מהורה
        if (t.GetRight() != null && t.GetLeft() == null
            && t.GetInfo() > t.GetRight().GetInfo())
            Console.WriteLine(t.GetInfo());
        PrintPernt(t.GetLeft());
        PrintPernt(t.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

מספר הצמתים בעץ

```
public static int NumNodes (BinTreeNode<int> bt)
{
    if (bt == null)
        return 0;
    return NumNodes(bt.GetLeft())+NumNodes(bt.GetRight())+1;
}
```

כיוון שהפעולה רקורסיבית, ההפניה יכולה להיות null בשלב כלשהו ברקורסיה

סופרים את הצמתים בתת עץ השמאלי

סופרים את הצמתים בתת עץ הימני

מוסיפים 1 עבור השורש שגם הוא צומת

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

תבנית מנייה

תבנית מנייה :

- תנאי עצירה הוא האם ההפניה היא null אז יוחזר 0
- אם התנאי מתקיים אז נעשה 1+קריאה רקורסיבת על תת-עץ-שמאל+קריאה רקורסיבית תת-עץ-ימני
- אם התנאי לא מתקיים נעשה קריאה רקורסיבת על תת-עץ-שמאל+קריאה רקורסיבית תת-עץ-ימני

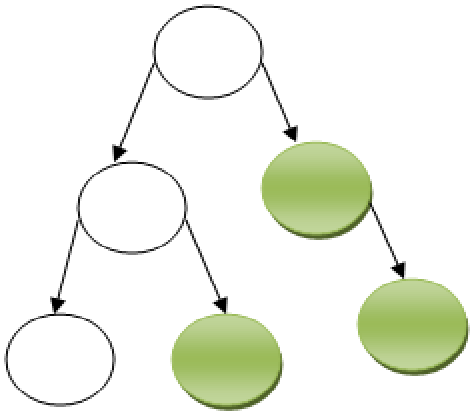
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמאות מנייה

```
//פעולה מחזירה מספר הצמתים הזוגיים בעץ //
public static int NumEven(BinTreeNode<int> t)
 {
        if (t == null)
            return 0;
        else if (t.GetInfo() % 2 == 0)
            return 1 + NumEven(t.GetLeft()) + NumEven(t.GetRight());
        else return NumEven(t.GetLeft()) + NumEven(t.GetRight());
  }
```

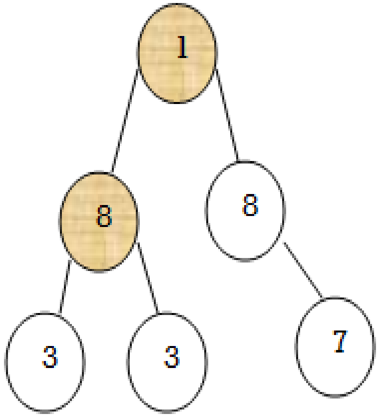
2.פעולה המחזירה מספר הבנים הימניים

```
//פעולה מחזירה מספר הבנים הימניים בעץ
public static int NumRightNode(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else if (t.GetRight() != null)
        return 1 + NumRightNode(t.GetLeft()) + NumRightNode(t.GetRight());
    else return NumRightNode(t.GetLeft());
}
```



3.פעולה המחזירה מספר הצמתים שיש להם 2 בנים שווים

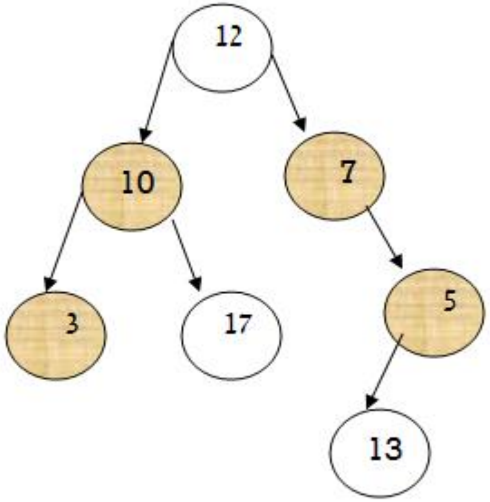
```
//פעולה מחזירה מספר הצמתים שיש להם 2 בנים ושווים
public static int NumFatherSonEqual(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else if (t.GetLeft()!= null && t.GetRight() != null &&
        t.GetLeft().GetInfo() == t.GetRight().GetInfo())
        return 1 + NumFatherSonEqual(t.GetLeft()) + NumFatherSonEqual(t.GetRight());
    else return NumFatherSonEqual(t.GetLeft()) + NumFatherSonEqual(t.GetRight());
}
```



4.פעולה מחזירה מספר הצמתים שערכם קטן מערכו של ההורה

```
//פעולה מחזירה מספר הצמתים שערכם קטן מערכו של ההורה שלהם
public static int NumNodeLessFather(BinTreeNode<int> t)
{
    int c,x;
    if (t == null)
        return 0;
    c = 0;
    x=t.GetInfo();
    if (t.GetLeft() != null && x > t.GetLeft().GetInfo())
        c++;
    if (t.GetRight() != null && x > t.GetRight().GetInfo())
        c++;
    return c + NumNodeLessFather(t.GetLeft()) + NumNodeLessFather(t.GetRight());

}
```



5.מספר העלים בעץ

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר//
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מחזירה מספר העלים בעץ//
public static int NumLeaf(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else if (IsLeaf(t))
        return 1;
    else return NumLeaf(t.GetLeft()) + NumLeaf(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

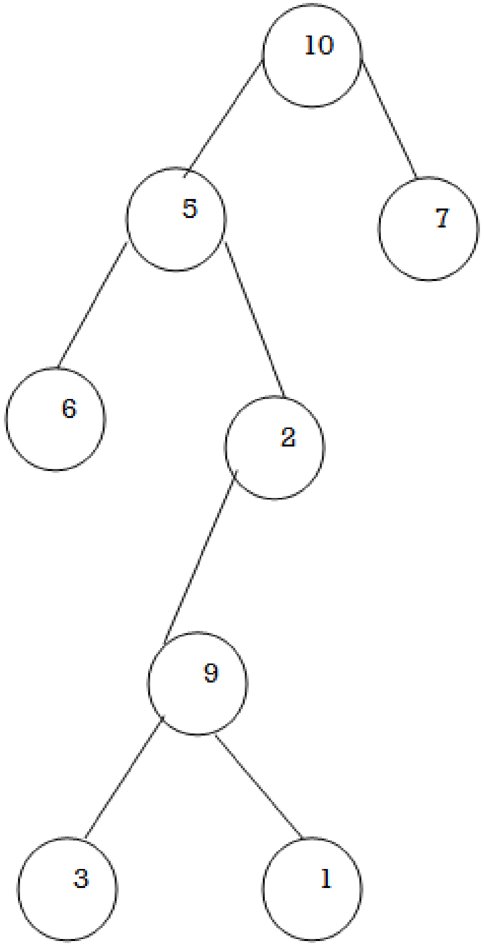
תבנית סכום צמתים בעץ

- אם ההפניה לעץ ריק נחזיר 0
- אם התנאי מתקיים,אז יש להחזיר **ערך הצומת**+קריאה רקורסיבית לתת-העץ-השמאלי+קריאה רקורסיבית לתת-העץ-הימני
- אם התנאי לא מתקיים אז יש להחזיר הצומת+קריאה רקורסיבית לתת-העץ-השמאלי+קריאה רקורסיבית לתת-העץ-הימני

דוגמה פתורה 1

```
//פעולה מחזירה סכום הצמתים בעץ
public static int SumNodes(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else return t.GetInfo() + SumNodes(t.GetLeft()) + SumNodes(t.GetRight());
}
```

דוגמה פתורה 2-סכום הסבא



עבור העץ הבא יוחזר 10+5+2=17

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמה פתורה 2-סכום הסבא

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר//
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מקבלת עץ ומחזירה את סכום הסאבים//
public static int SumGrandfather(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else if (t.GetLeft() != null && !IsLeaf(t.GetLeft())||
             t.GetRight()!=null && !IsLeaf(t.GetRight()))
        return t.GetInfo() + SumGrandfather(t.GetLeft()) + SumGrandfather(t.GetRight());
    else return SumGrandfather(t.GetLeft()) + SumGrandfather(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

דוגמה פתורה 3 –סכום צמתים ברמה

```
//פעולה מקבלת עץ ומחזירה אמת אם כל הצמתים הם זוגיים
public static int SumLevel(BinTreeNode<int> t,int level)
{
    if (t == null)
        return 0;
    if (level == 0)
        return t.GetInfo();
    return SumLevel(t.GetLeft(), level - 1) + SumLevel(t.GetRight(), level - 1);
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

תבניות לבעיות שמחזירות ערך בוליאני

- תבנית זו דנה בהגדרת עץ המקיים תנאי מסויים,אם התנאי מתקיים תחזיר הפעולה אמת אחרת תחזיר שקר

1. בהתחלה יש לבדוק את תנאי העצירה ולהחזיר ערך מתאים
2. לאחר בדיקת תנאי העצירה,יש לבדוק את כל המקרים בהם **לא מתקיים** התנאי ולהחזיר שקר
3. יש להחזיר את **תוצאות הקריאה הרקורסיבית** המתאימה

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בדיקת קיום איבר בעץ

```
public static bool Exists (BinTreeNode<int> bt, int x)
{
    if (bt == null)
        return false;

    if (bt.GetInfo() == x)
        return true;

    return Exists(bt.GetLeft(), x) || Exists(bt.GetRight(), x);
}
```

תנאי העצירה

סריקה בסדר תחילי (אפשר היה להשתמש בכל סריקה אחרת)

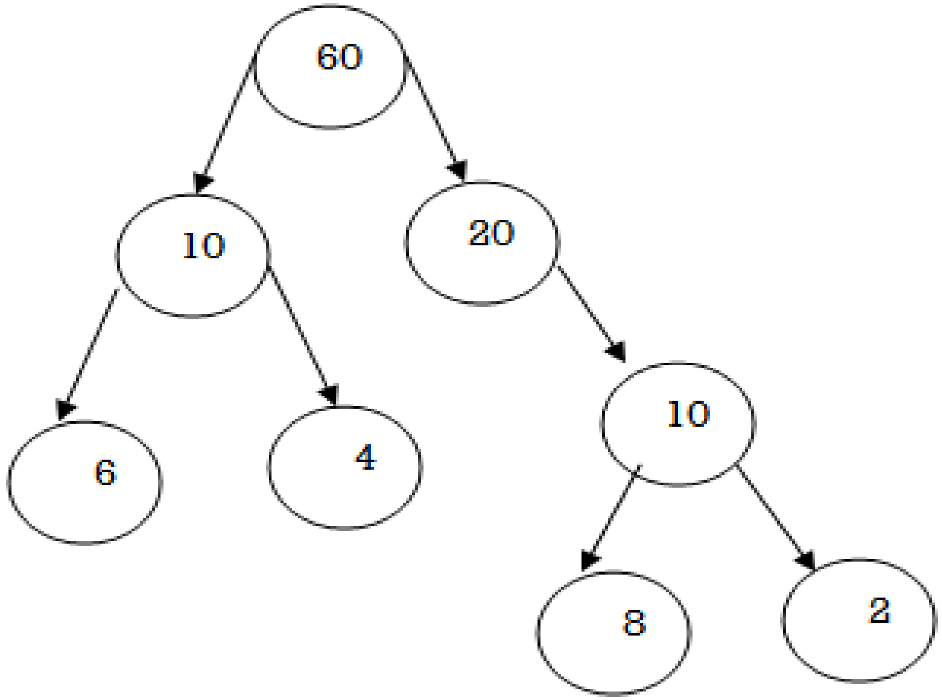
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

2.האם כל הצמתים בעץ הם זוגיים?

```
//פעולה מקבלת עץ ומחזירה אמת אם כל הצמתים הם זוגיים//
public static bool IsAllNodeEven(BinTreeNode<int> t)
{
    if (t == null)
        return true;
    else if (t.GetInfo() % 2 != 0)
        return false;
    else return IsAllNodeEven(t.GetLeft()) && IsAllNodeEven(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ סכומים הוא עלה או עץ שבו כל צומת הוא סכום כל הצאצאים



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ סכומים המשך האם הפתרון נכון?? הסבירו!

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}

//פעולה מחזירה סכום הצמתים בעץ
public static int SumNodes(BinTreeNode<int> t)
{
    if (t == null)
        return 0;
    else return t.GetInfo() + SumNodes(t.GetLeft()) + SumNodes(t.GetRight());
}
//פעולה מקבלת עץ ומחזירה אמת אם העץ הוא עץ סכומים אחרת יוחזר שקר
public static bool IsTreeSum(BinTreeNode<int> t)
{
    if (IsLeaf(t))
        return true;
    if (t.GetInfo() != SumNodes(t.GetLeft())+SumNodes(t.GetRight()))
        return false;
    return IsTreeSum(t.GetLeft()) && IsTreeSum(t.GetRight());
}
```

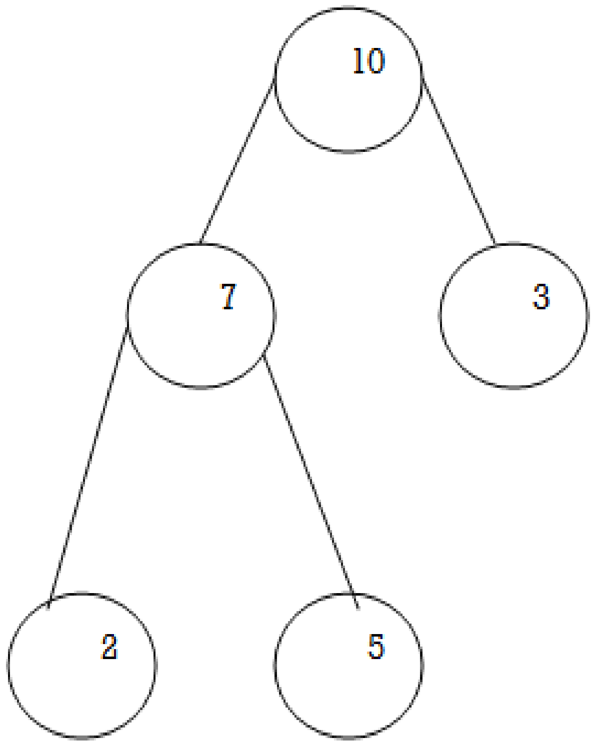

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

3.האם לכל צומת (לא עלה) יש שני בנים?

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מקבלת עץ ומחזירה אמת אם לכל צוממת יש שני ילדים
public static bool EachHasTwoChild(BinTreeNode<int> t)
{
    if (IsLeaf(t))//אם הצומת עלה
        return true;
    else if (t.GetLeft() == null || t.GetRight() == null)
        return false;
    else return EachHasTwoChild(t.GetLeft()) && EachHasTwoChild(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

4. עץ-שווה בנים-הוא עץ בינארי לא ריק,
ערך השורש שווה לסכום ערכי בניו וכל אחד מהבנים הוא עץ-שווה-בנים



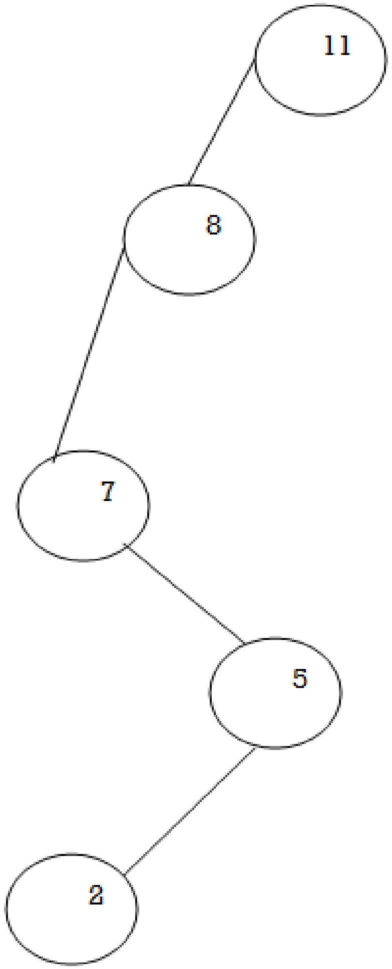
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

4.פתרון שאלה "עץ-שווה-בנים"

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מקבלת עץ ומחזירה אמת אם הוא עלה או שערך שורשו שווה לסכום ערכי בניו
//וגם כל אחד מבניו מקיים את התכונה
public static bool EqualTowChild(BinTreeNode<int> t)
{
    if (IsLeaf(t))//אם צומת עלה
        return true;
    else if (t.GetLeft() == null || t.GetRight() == null)
        return false;
    else if (t.GetInfo() != t.GetLeft().GetInfo() + t.GetRight().GetInfo())
        return false;
    else return EqualTowChild(t.GetLeft()) && EqualTowChild(t.GetRight());
}
```

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

5.עץ יורד הוא עלה או שורש ובן יחיד,כך שערך הבן קטן מערך השורש והבן עץ-יורד



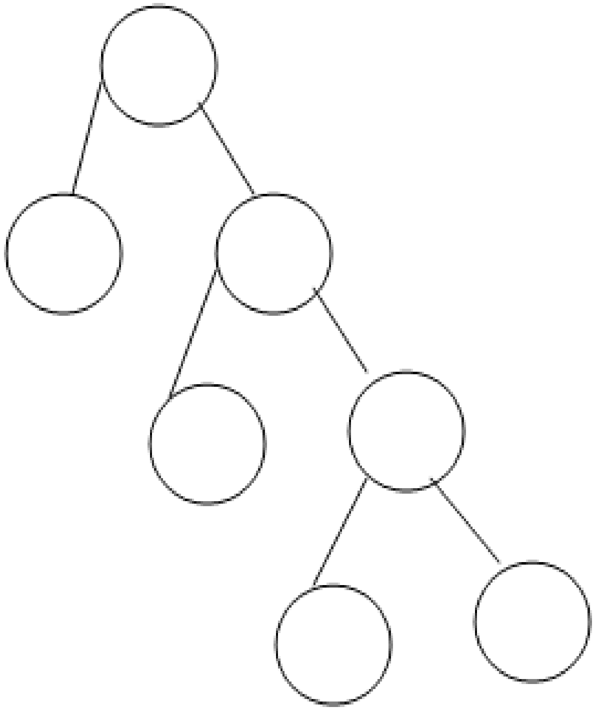
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

5.פתרון שאלה עץ -יורד

```
//פעולה מקבלת עץ ובודקת האם עץ הוא עץ יורד
//עץ יורד הוא עלה או שורש ובן אחד,כך  שערך הבן אינו גדול מערך השורש
//והבן עץ יורד
public static bool IsTreeDown(BinTreeNode<int> t)
{
    if (t.GetLeft() == null && t.GetRight() == null)
        return true;//עלה
    else if (t.GetLeft() != null && t.GetRight() != null)
        return false;//האם קיים 2 בנים
    else if (t.GetLeft() != null && t.GetInfo() < t.GetLeft().GetInfo())
        return false;
    else if (t.GetRight() != null && t.GetInfo() < t.GetRight().GetInfo())
        return false;
    else if (t.GetLeft() != null)
        return IsTreeDown(t.GetLeft());
    else return IsTreeDown(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

"עץ_משולשים_ימניים" הוא עץ שבו צומת אחד, או עץ שבו לכל ילד ימני של צומת יש שני ילדים, וכל ילד שמאלי הוא עלה (כמתואר בשרטוט).



© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

5.פתרון שאלה עץ-משולש ימניים

```
//פעולה מקבלת עץ ומחזירה אמת אם העץ מהווה עץ משולש ימני ואחרת שקר
public static bool RightTreeTriangle(BinTreeNode<int> t)
{
    if (IsLeaf(t)) //אם צומת עלה
        return true;
    if (t.GetLeft() == null || t.GetRight() == null)//בן יחיד
        return false;
    if (!IsLeaf(t.GetLeft()))//אם בן שמאלי לא עלה
        return false;
    return RightTreeTriangle(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

6.עץ נקרא דור-שלשי הוא עץ לא ריק,אם קיים בעץ נכדים(נכד הוא בן של בן)

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מקבלת עץ ומחזירה אמת אם קיים בעץ נכדים//
public static bool IsGrandson(BinTreeNode<int> t)
{
    if (t==null|| IsLeaf(t))//אם הצומת עלה
        return false;
     if (t.GetLeft()!=null && !IsLeaf(t.GetLeft())||
         t.GetRight()!= null && !IsLeaf(t.GetRight()))
         return true;
     return IsGrandson(t.GetLeft()) || IsGrandson(t.GetRight());
}
```

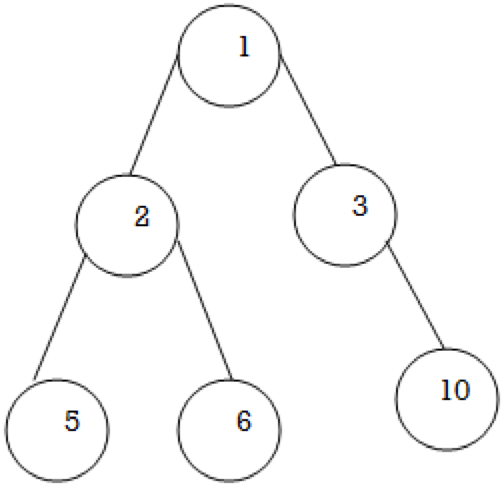

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

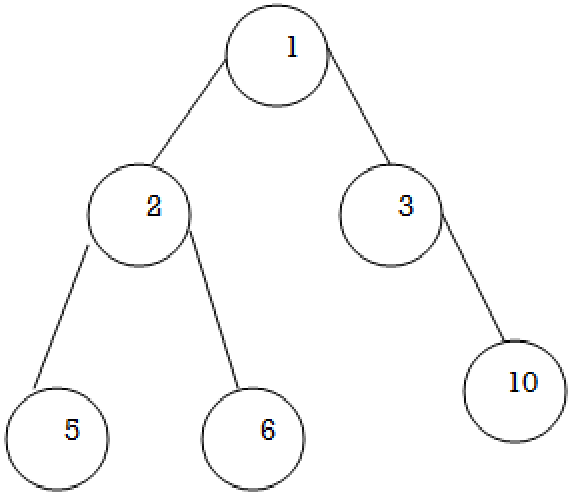
7.דוגמה נוספת.

```
//פעולה מקבלת עץ ומחזירה אמת אם קיימים בעץ שני עלים שהם אחים//
//ושקר אחרת//
public static bool IsTwoChildLeaf(BinTreeNode<int> t)
{
    if (t==null|| IsLeaf(t))//אם צומת עלה או עץ ריק
        return false;
    if (t.GetLeft()!=null && IsLeaf(t.GetLeft()) &&
        t.GetRight()!=null && (IsLeaf(t.GetRight())))
        return true;
    return IsTwoChildLeaf(t.GetLeft()) || IsTwoChildLeaf(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

8. פעולה המקבלת 2 עצים לא ריקים ומחזירה אמת אם העץ הראשון זהה בתוכן וגם במבנה לעץ השני-דוגמה





© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

פתרון עצים דומים

```
//פעולה מקבלת 2 עצים ומחזירה אמת אם שני העצים דומים בתוכן וגם במבנה
//הנחה 2 העצים לא ריקים
public static bool IsEqual(BinTreeNode<int> t1, BinTreeNode<int> t2)
{
    if (t1 == null && t2 == null)
        return true;
    if (t1.GetInfo() != t2.GetInfo())
        return false;
    if (t1.GetLeft() != null && t2.GetLeft() == null)
        return false;
    if (t1.GetLeft() == null && t2.GetLeft() != null)
        return false;
    if (t1.GetRight() == null && t2.GetRight() != null)
        return false;
    if (t1.GetRight() != null && t2.GetRight() == null)
        return false;
    return IsEqual(t1.GetLeft(), t2.GetLeft()) &&
           IsEqual(t1.GetRight(), t2.GetRight());
}
```

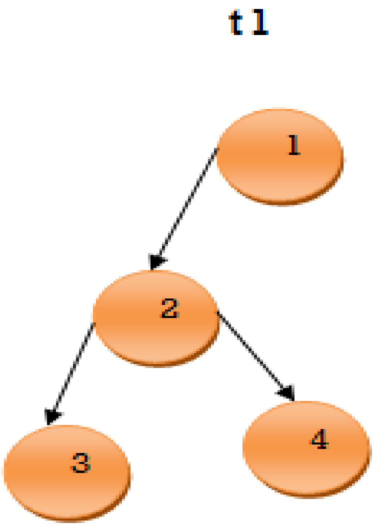

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

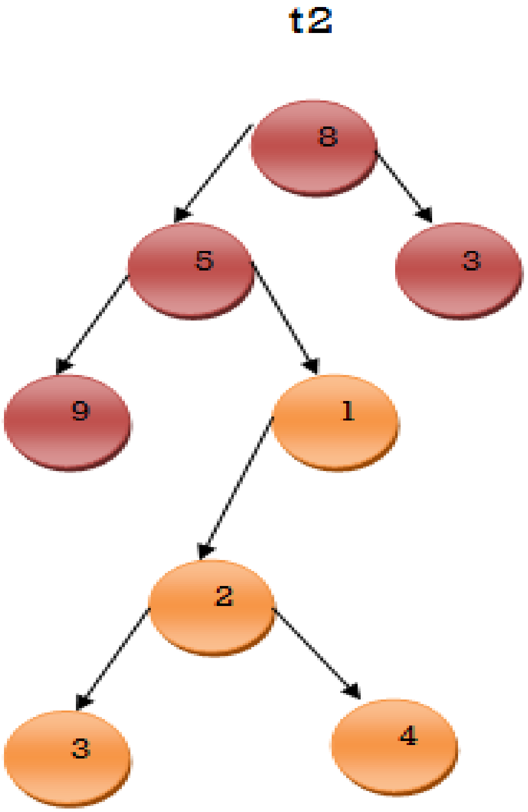
עץ תעלומה הוא עלה או לכל צומת
יש שני ילדים כך שכל בן שמאלי הוא שללי וכל בן ימני הוא חיובי
כתוב פעולה הבודקת אם עץ הוא תעלומה,

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//פעולה מקבלת עץ ומחזירה אמת אם העץ הוא תעלומה ושקר אחרת
public static bool IsTaloma(BinTreeNode<int> t)
{
    if (IsLeaf(t))//אם עלה
       return true;
    if (t.GetLeft() == null || t.GetRight() == null)
        return false;
    if (t.GetLeft().GetInfo() > 0)
        return false;
    if (t.GetRight().GetInfo() < 0)
        return false;
    return IsTaloma(t.GetLeft()) && IsTaloma(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ מוכל? (האם עץ t1 מוכל בעץ t2)





© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ מוכל? (האם עץ t1 מוכל בעץ t2)

```
//פעולה מקבלת 2 עצים ומחזירה אמת אם שני העצים דומים בתוכן וגם במבנה
public static bool IsEqual(BinTreeNode<int> t1, BinTreeNode<int> t2)
{
    if (t1 == null && t2 == null)
        return true;
    if (t1.GetInfo() != t2.GetInfo())
        return false;
    if (t1.GetLeft() != null && t2.GetLeft() == null)
        return false;
    if (t1.GetLeft() == null && t2.GetLeft() != null)
        return false;
    if (t1.GetRight() == null && t2.GetRight() != null)
        return false;
    if (t1.GetRight() != null && t2.GetRight() == null)
        return false;
    return IsEqual(t1.GetLeft(), t2.GetLeft()) &&
           IsEqual(t1.GetRight(), t2.GetRight());
}
//פעולה מקבלת 2 עצים ומחזירה אמת אם העץ t1 מוכל בתוך העץ t2
public static bool IsContained(BinTreeNode<int> t1, BinTreeNode<int> t2)
{
    if (t1 == null)
        return true;
    if (t2 == null)
        return false;
    if (IsEqual(t1, t2))
        return true;
    return IsContained(t1, t2.GetLeft()) || IsContained(t1, t2.GetRight());
}
```

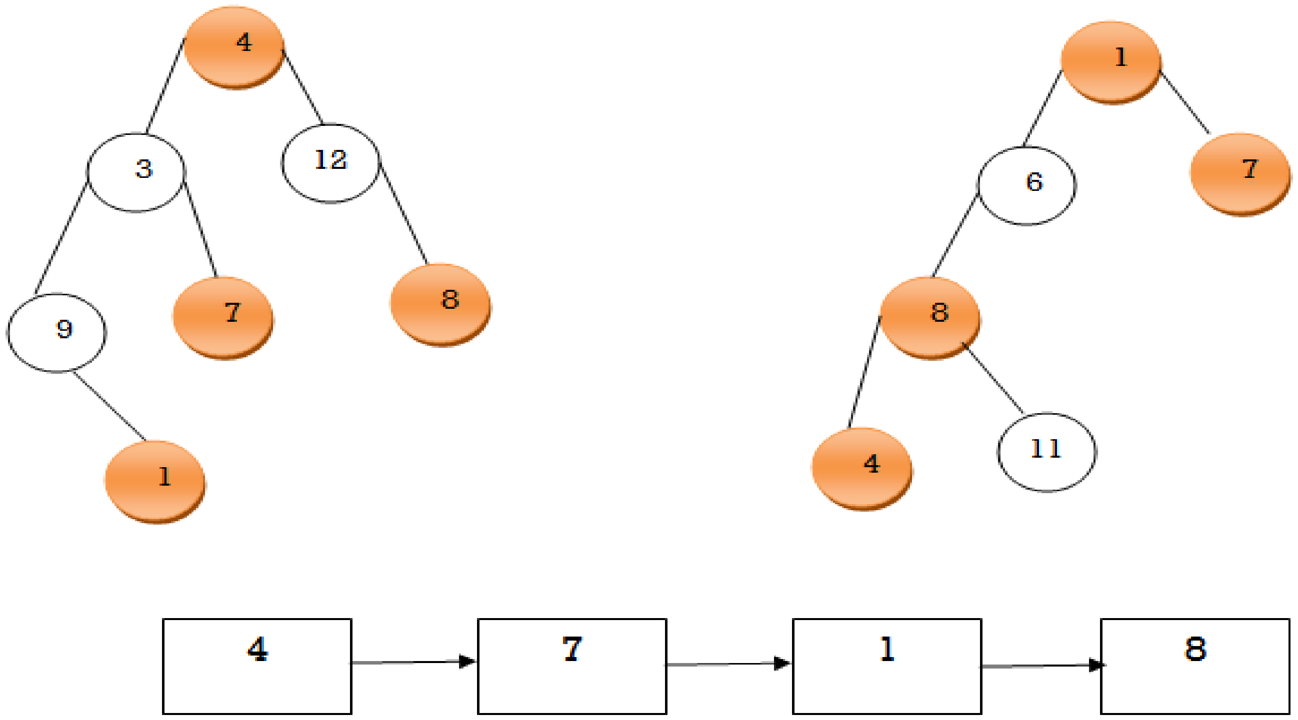

האוניברסיטה העברית בירושלים

דוגמאות לפעולות כלליות

```
//פעולה מחזיר את האיבר המקסימאלי בין 3 מספרים//
public static int Max3(int x, int y, int z)
{
    return Math.Max(Math.Max(x, y), z);
}
//פעולה מקבלת עץ ומחזירה את האיבר המקסימאלי בעץ//
public static int GetMax(BinTreeNode<int> t)
{
    if (IsLeaf(t))//צומת עלה
        return t.GetInfo();
    if (t.GetLeft() == null)
        return Math.Max(t.GetInfo(),GetMax(t.GetRight()));
    if (t.GetRight() == null)
        return Math.Max(t.GetInfo(), GetMax(t.GetLeft()));
    return Max3(t.GetInfo(),GetMax(t.GetLeft()),GetMax(t.GetRight()));
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

פעולה המקבלת 2 עצים ומחזירה רשימה חדשה המכילה האיברים המשותפים ל 2 עצים.(איברי העצים שונים זה מזה)



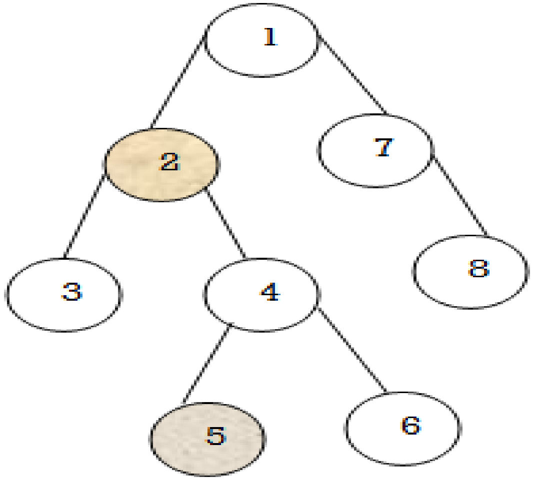
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

```
//פעולה מחזירה אמת אם מספר נמצא בעץ אחרת מחזירה הפעולה שקר
public static bool Found(BinTreeNode<int> t, int X)
{
    if (t == null)
        return false;
    if (t.GetInfo() == X)
        return true;
    return Found(t.GetLeft(), X) || Found(t.GetRight(), X);
}
//פעולה מקבלת 2 עצים לא ריקים ובונה רשימת האיברים המשותפים לשני העצים
public static void BuildList(BinTreeNode<int> t1, BinTreeNode<int> t2, List<int> lst)
{
    if (t1 != null)
    {
        if (Found(t2, t1.GetInfo()))
            lst.Insert(null, t1.GetInfo());
        BuildList(t1.GetLeft(), t2, lst);
        BuildList(t1.GetRight(), t2, lst);
    }
}
//פעולה מקבלת 2 עצים לא ריקים ומחזירה רשימה של האיברים המשותפים
public static List<int> CommonNodes(BinTreeNode<int> t1, BinTreeNode<int> t2)
{
    List<int> list = new List<int>();
    BuildList(t1, t2, list);
    return list;
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

צאצא

נניח כי x=5 ו-y=2 נאמר כי x הוא צאצא של y




```
// פעולה מקבלת עץ ומחזירה אמת אם קיים צומת x והוא צאצא של צומת y ושקר אחרת
public static bool Descendant(BinTreeNode<int> t,int x,int y)
{
    if (t.GetInfo() == y)
        return Found(t.GetLeft(), x) || Found(t.GetRight(), x);
    return Descendant(t.GetLeft(), x, y) || Descendant(t.GetRight(), x, y);
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

מחרוזת המתארת את העץ

```
public static string PreorderString (BinTreeNode<string> bt)
{
    if (bt == null)
        return "";
    return bt.GetInfo() + " " + PreorderString (bt.GetLeft()) +
                                PreorderString (bt.GetRight());
}
```

ממשו את הפעולות PostorderString(…)
ו-InorderString(…)

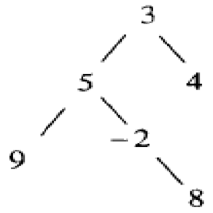
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות 2005

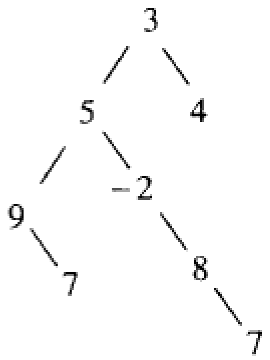
.3 לפניך פעולה:

הוסף_עלים (T , N)	הפעולה מקבלת מספר שלם N ועץ בינארי T לא ריק שערכיו הם מספרים שלמים. לכל עלה בעץ T שערכו גדול מ- N, הפעולה מוסיפה בן ימני שערכו N. הנחה: העץ T מאותחל.

דוגמה: נתון העץ T



לאחר זימון הפעולה **הוסף_עלים (T , 7)**, העץ T הוא:



כתוב אלגוריתם שיממש את הפעולה **הוסף_עלים (T , N)**.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות 2005 המשך פתרון

```
//פעולה מחזירה אמת אם הצומת עלה אחרת תחזיר הפעולה שקר
public static bool IsLeaf(BinTreeNode<int> bt)
{
    return (bt.GetLeft() == null) && (bt.GetRight() == null);
}
//בגרות 2005
//פעולה מקבלת עץ לא ריקה ומספר n לכל עלה בעץ שערכו גדול מ- n הפעולה מוסיפה בן ימני
//שערכו שווה ל n
public static void InsertLeaf(BinTreeNode<int> t, int n)
{
    if (t != null)
    {
        if (IsLeaf(t) && t.GetInfo()>n)
        {
            t.SetRight(new BinTreeNode<int>(n));
        }
        InsertLeaf(t.GetLeft(),n);
        InsertLeaf(t.GetRight(),n);
    }
}
```

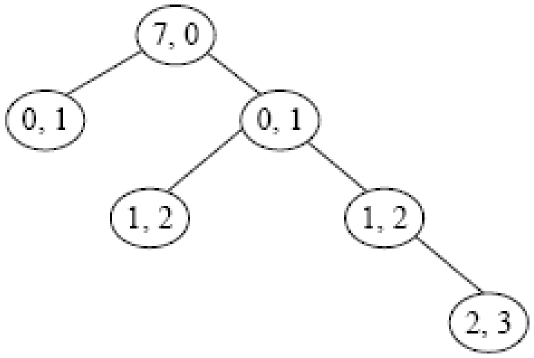

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות 2005 מועד מיוחד

3. לפניך פעולה:

רמה_בבנים (T)	הפעולה מקבלת עץ בינארי T. בכל צומת בעץ שני ערכים – האחד הוא מספר שלם שהוא ערך הצומת, והאחר מציין את רמת הצומת. הפעולה מחזירה 'אמת' אם לכל צומת בעץ, פרט לשורש העץ, מתקיים שערך הצומת הוא הרמה של אביו.

לפניך העץ T:



עבור העץ T הפעולה מחזירה 'אמת'.

כתוב אלגוריתם שיממש את הפעולה **רמה_בבנים** (T).

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות מועד מיוחד 2005 המשך פתרון (נכתוב מחלקה חדשה)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication1
{
    public class InfoLevel
    {
        private int root;
        private int level;
        public InfoLevel(int root, int level)
        {
            this.root = root;
            this.level = level;
        }
        public int GetRoot()
        {
            return root;
        }
        public int GetLevel()
        {
            return level;
        }
        public override string ToString()
        {
            return root + "," + level;
        }
    }
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות מועד מיוחד 2005 המשך פתרון

```
//בגרות 2005 מועד מיוחד
//הפעולה מקבלת עץ ומחזירה אמת אם העץ "רמה_בבנים" אחרת תחזיר הפעולה שקר
//הנחה עץ לא ריק
public static bool LevelChild(BinTreeNode<InfoLevel> t)
{
    if (t==null)
        return true;
    if (t.GetLeft() != null && t.GetLeft().GetInfo().GetRoot()!=t.GetInfo().GetLevel())
        return false;
    if (t.GetRight() != null && t.GetRight().GetInfo().GetRoot() != t.GetInfo().GetLevel())
        return false;
    return LevelChild(t.GetLeft()) && LevelChild(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

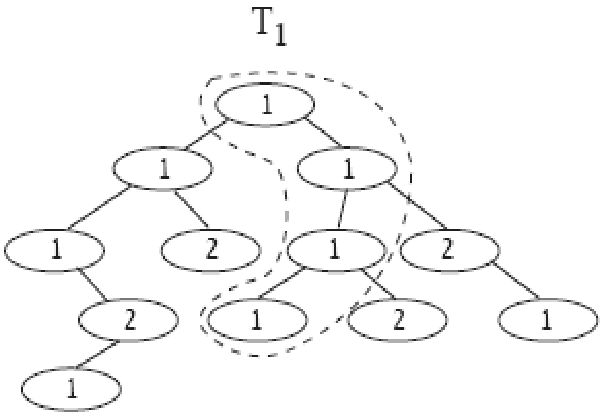
בגרות 2007

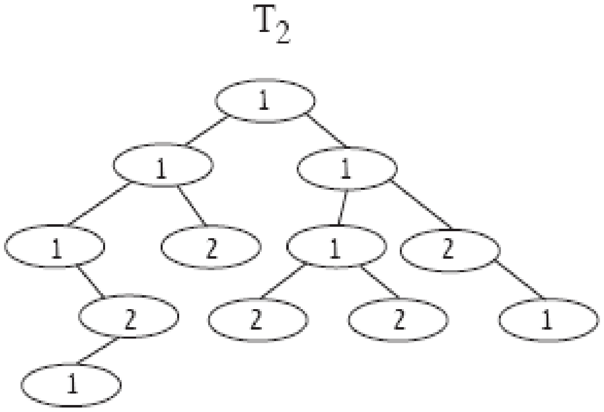
3. עץ "דו־מספרי" הוא עץ בינרי לא ריק, שהערכים בצמתים שלו הם המספרים 1 או 2.

על עץ "דו־מספרי" מוגדרת פעולה **מסלול-אחיד** המחזירה "אמת", אם קיים בעץ מסלול, המתחיל בשורש העץ ומסתיים באחד העלים שלו, וכל ערכי הצמתים בו זהים. אם לא קיים מסלול כזה, הפעולה מחזירה "שקר".

דוגמאות: בעבור העץ T1 הפעולה **מסלול-אחיד** תחזיר "אמת".

בעבור העץ T2 הפעולה **מסלול-אחיד** תחזיר "שקר".





© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בגרות 2007 פתרון

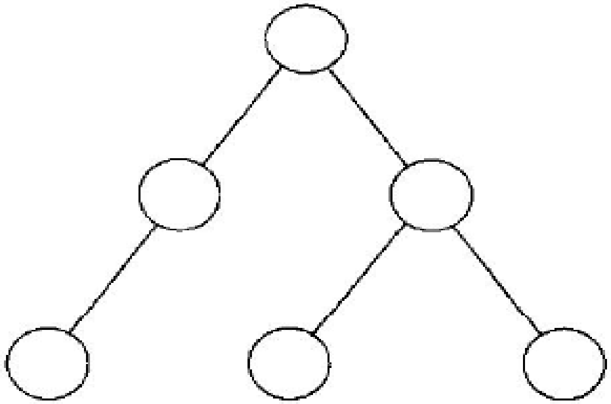
```
//פעולה מחזירה אמת אם הצומת עלה אחרת הפעולה תחזיר שקר
public static bool IsLeaf(BinTreeNode<int> t)
{
    return (t.GetLeft() == null) && (t.GetRight() == null);
}
//בגרות 2007
//פעולה מקבלת עץ ומחזירה אמת אם קיים "מסלול-אחיד" אחרת הפעולה תחזיר שקר
//הנחה:עץ לא ריק
public static bool OnePath(BinTreeNode<int> t)
{
    if (t == null)
        return false;
    if (IsLeaf(t))//אם צומת עלה
        return true;
    if (t.GetLeft()==null && t.GetRight()!=null && t.GetInfo()!=t.GetRight().GetInfo())
        return false;
    if (t.GetRight() == null && t.GetLeft() != null && t.GetInfo() != t.GetLeft().GetInfo())
        return false;
    if (t.GetLeft() != null && t.GetRight() != null &&
        t.GetLeft().GetInfo() != t.GetInfo() && t.GetRight().GetInfo() != t.GetInfo())
        return false;
    return OnePath(t.GetLeft()) || OnePath(t.GetRight());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

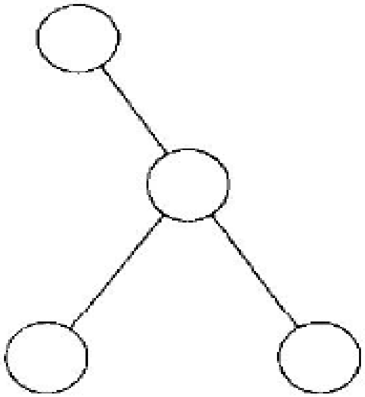
בגרות 2004

.2 עץ בינרי T הוא עץ "פרו ורבו", אם קיים **בו** צומת אשר יש לו לפחות שני נכדים, וכל נכד מבן אחר. <u>הערה</u>: נכד הוא רק בן של בן.

<u>דוגמה</u>



עץ "פרו ורבו"



לא עץ "פרו ורבו"

כתוב אלגוריתם **האם_פרו_ורבו!**(T) , המקבל עץ בינרי T ומחזיר 'אמת' אם הוא עץ "פרו ורבו", ואחרת – 'שקר'.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

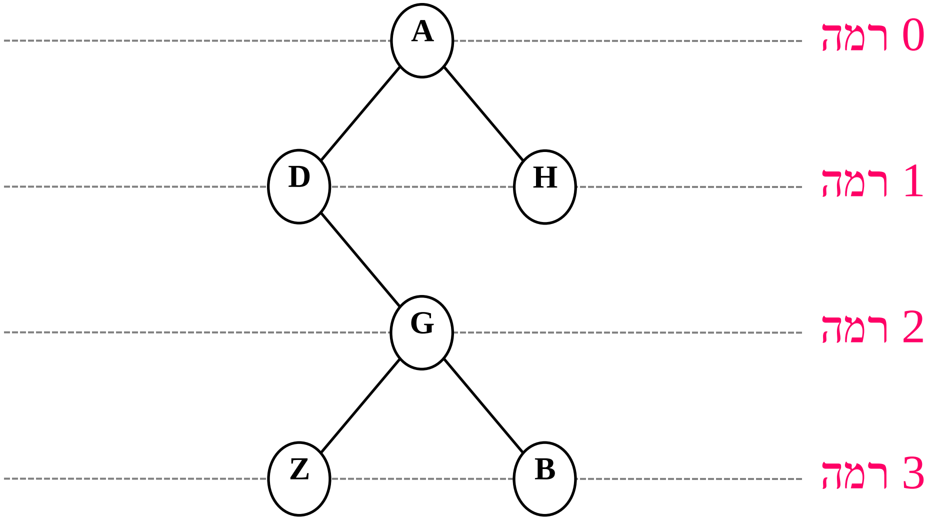
בגרות 2004 פתרון

```
//פעולה מקבלת עץ ומחזירה אמת אם העץ פרו ורבו אחרת יוחזר שקר
public static bool ProRbo(BinTreeNode<int> t)
{
    if (t == null)
        return false;
    if (t.GetLeft() != null && !IsLeaf(t.GetLeft()) &&
        t.GetRight() != null && !IsLeaf(t.GetRight()))
        return true;
    return ProRbo(t.GetLeft()) || ProRbo(t.GetLeft());
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סריקה לפי רמות

רמות של עץ:



איך אפשר לסרוק לפי רמות ולהדפיס:

ADHGZB?

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סריקה לפי רמות

הדפס-לפי-רמות (tree)

בנה תור חדש של חוליות בינריות

הכנס את השורש לתוך התור

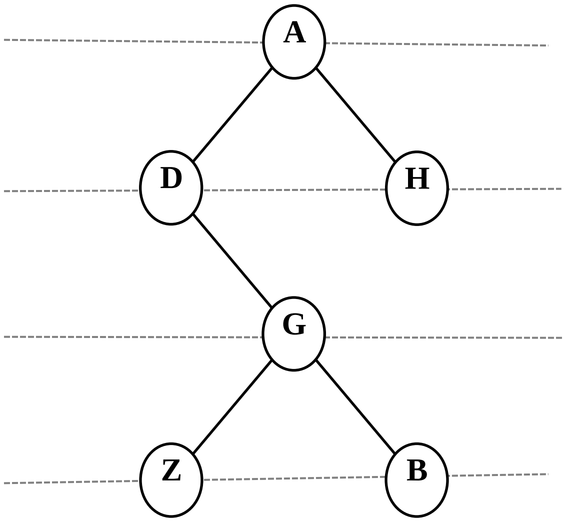
כל עוד התור אינו ריק, בצע את הפעולות:

הוצא חוליה מתוך התור

הדפס את ערך החוליה

אם קיים לחוליה ילד שמאלי, הכנס אותו לתור

אם קיים לחוליה ילד ימני, הכנס אותו לתור



G HB AZD

A D H G Z B

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סריקה לפי רמות

```
//פעולה מקבלת עץ ומדפיסה את הצמתים לפי רמות
public static void LevelOrder(BinTreeNode<int> t)
{
    Queue<BinTreeNode<int>> q = new Queue<BinTreeNode<int>>();
    BinTreeNode<int> bt;
    q.Insert(t);
    while (!q.IsEmpty())
    {
        bt = q.Remove();
        Console.WriteLine(bt.GetInfo());
        if (bt.GetLeft() != null)
            q.Insert(bt.GetLeft());
        if (bt.GetRight() != null)
            q.Insert(bt.GetRight());
    }
}
```


© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

```
//פעולה מקבלת עץ הפעולה מחזירה סכום הערכים של כל רמה בעץ לרשימה חדשה
public static List<int>  LevelOrderSumList(BinTreeNode<int> t)
{
    List<int> list = new List<int>();
    Node<int> pos = list.GetFirst();
    Queue<BinTreeNode<int>> q1 = new Queue<BinTreeNode<int>>();
    Queue<BinTreeNode<int>> q2 = new Queue<BinTreeNode<int>>();
    BinTreeNode<int> bt;
    q1.Insert(t);
    while (!q1.IsEmpty() || !q2.IsEmpty())
    {
        int sum = 0;
        if (!q1.IsEmpty())
            while (!q1.IsEmpty())
            {
                bt = q1.Remove();
                sum = sum + bt.GetInfo();
                if (bt.GetLeft() != null)
                    q2.Insert(bt.GetLeft());
                if (bt.GetRight() != null)
                    q2.Insert(bt.GetRight());
            }
        else
            while (!q2.IsEmpty())
            {
                bt = q2.Remove();
                sum = sum + bt.GetInfo();
                if (bt.GetLeft() != null)
                    q1.Insert(bt.GetLeft());
                if (bt.GetRight() != null)
                    q1.Insert(bt.GetRight());
            }
        pos = list.Insert(pos, sum);
    }
    return list;
}
```

שאלה

- ממשו פעולה בשם LevelOrderString המקבלת עץ חוליות בינרי של מחרוזות ומחזירה מחרוזת המתארת את תוכן העץ המסודר לפי רמות העץ.
- כדי לכתוב את הקוד יהיה עליכם להגדיר תור של חוליות בינריות מטיפוס מחרוזת, כלומר, הגנריות תכתב עבור החוליה הבינרית ועבור התור בו זמנית.

ההגדרה הגנרית הכפולה הזו תיראה כך:

Queue<BinTreeNode<string>>

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

שימוש בעץ חוליות בינרי – בדיקת תקינות של ביטוי חשבוני

על מנת לפשט את הבעיה :

- מספרים חד ספרתיים בלבד – אופרנדים.
- 4 פעולות חשבון (חיבור, חיסור, כפל וחילוק) -אופרטורים.
- אין מספרים שליליים.
- הביטוי "ממוסגר לחלוטין".

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

ביטוי חשבוני

ביטויים חוקיים :

(7 + 5)

((3 * 4) + 2)

((3 – 2) * ((4 * 1) + 8))

ביטויים לא חוקיים :

(3 + 5) * 4)

2 – 3

(–7)

(8)

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

ביטוי חשבוני

- **ביטוי חשבוני** הוא:
- **A** כאשר A הוא אופרנד (מספר חד ספרתי)

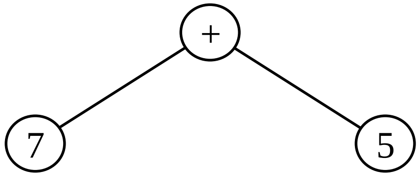
או: **(X op Y)** כאשר X ו-Y הם ביטויים חשבוניים, האופרנדים של הפעולה, ו-op הוא פעולה.

הגדרה רקורסיבית, דומה להגדרה של עץ

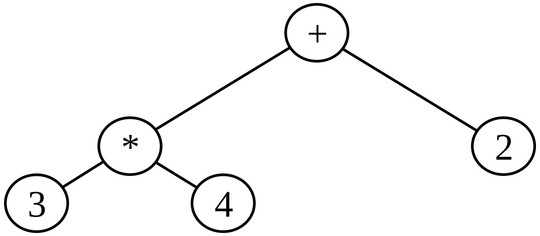
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

ביטוי חשבוני

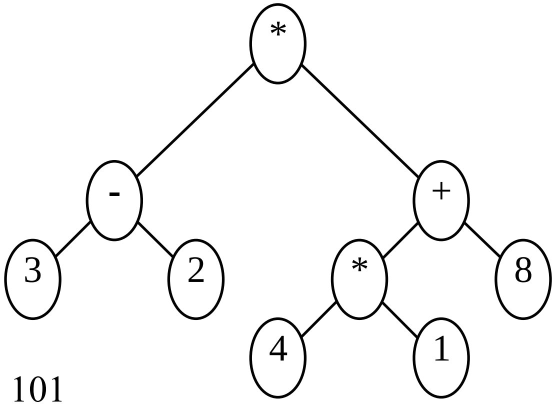
אופרנדים – רק בעלים :



- הביטוי (7 + 5):



- הביטוי ((3 * 4) + 2)



- הביטוי ((3 – 2) * ((4 * 1) + 8))

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

שאלה

כיצד ייוצג הביטוי:

((4 + (7 * 8)) – (9 / 3))?

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

חישוב ערך הביטוי החשבוני

חשב-ערך-ביטוי (tree)

אם העץ הוא עלה, החזר את ערכו

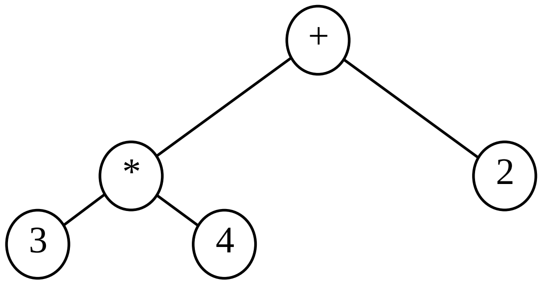
אחרת,

***חשב-ערך-ביטוי** (**תת עץ שמאלי של** tree) ושמור את התוצאה ב- leftVal.*

***חשב-ערך-ביטוי** (**תת עץ ימני של** tree) ושמוראת התוצאה ב- rightVal.*

שמור ב-op את הערך של השורש

החזר את: (leftVal op rightVal).



בצעו מעקב של האלגוריתם על העץ הנתון:

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

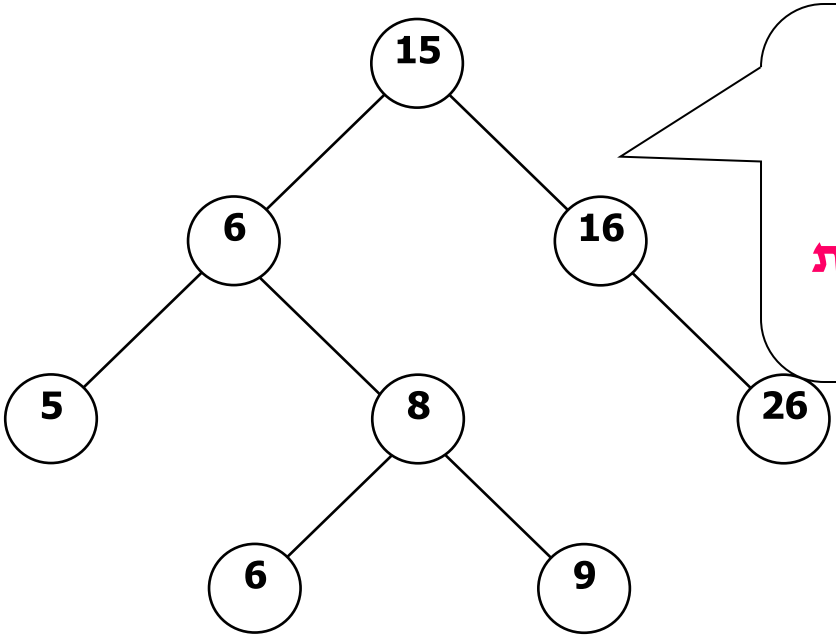
שאלה

? ממשו את האלגוריתם **חשב-ערך-ביטוי**

כפעולה בשם ComputeExprTree(…)

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ-חיפוש-בינרי (Binary Search Tree)



כל הערכים הנמצאים בתת-עץ השמאלי של צומת כלשהו קטנים מהערך שבצומת, וכל הערכים הנמצאים בתת-עץ הימני של הצומת גדולים או שווים לערך זה.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

איתור ערך בעץ-חיפוש-בינרי

```
public static bool ExistsInBST (BinTreeNode<int> bst, int x)
{
    if (bst == null)
        return false;

    if (bst.GetInfo() == x)
        return true;

    if (x < bst.GetInfo())
        return ExistsInBST (bst.GetLeft(), x);

    return ExistsInBST (bst.GetRight(), x);
}
```

הפרמטר הוא
עץ-חיפוש-בינרי

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

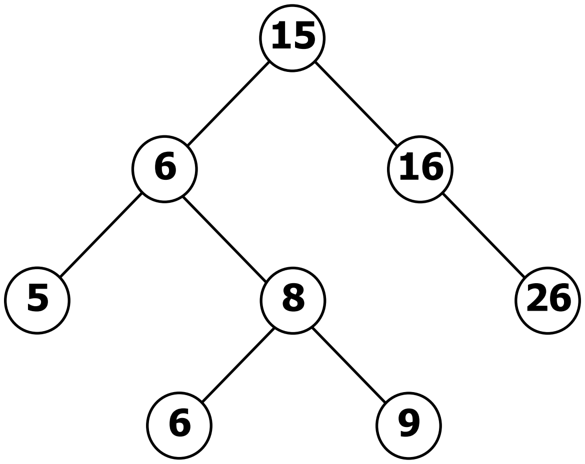
שאלה

כתבו גרסה איטרטיבית ?

של הפעולה ExistsInBST(...)

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

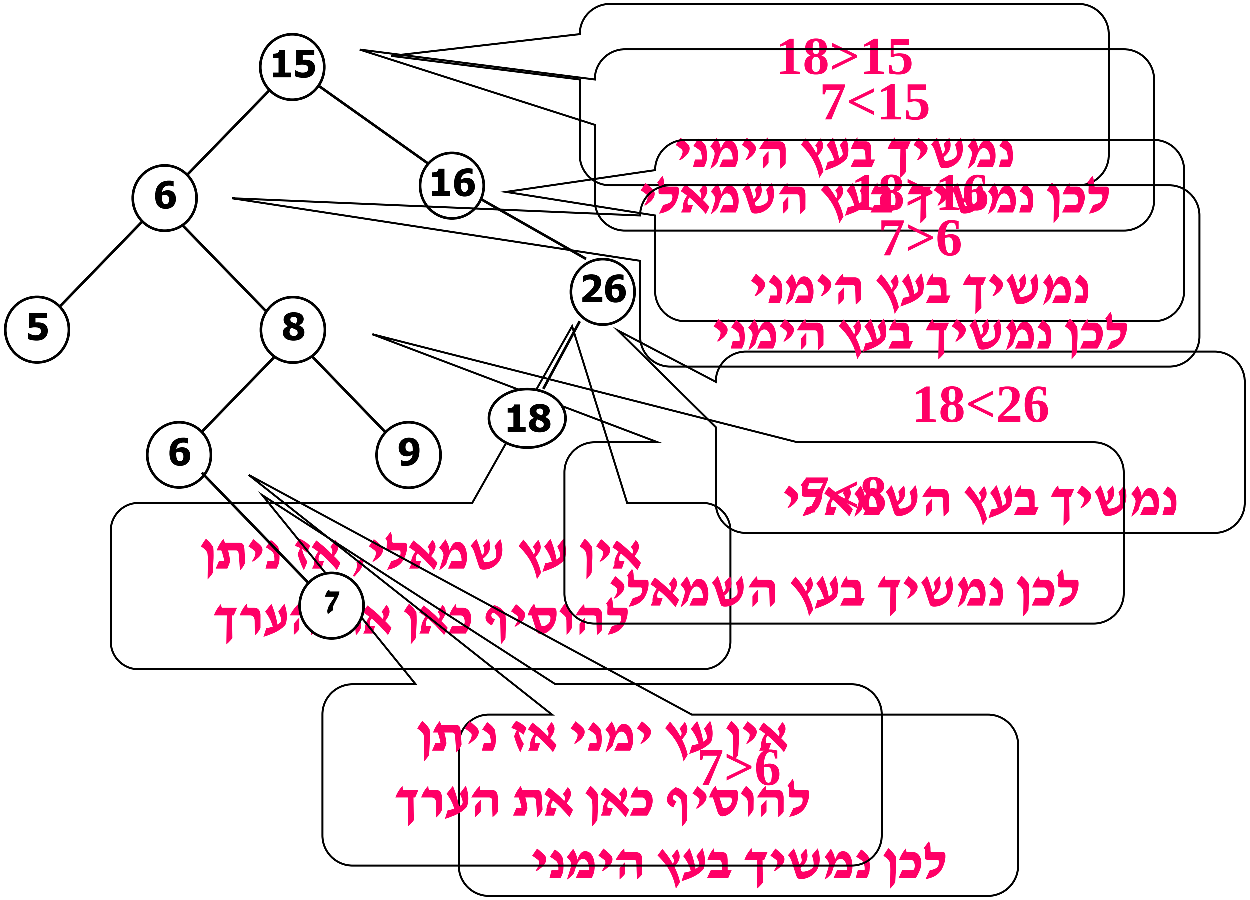
מציאת ערך מינימלי בעץ-חיפוש-בינרי



- היכן נמצא הערך הקטן ביותר?
- האם תמיד הערך הקטן ביותר נמצא בצומת השמאלי ביותר? נמקו.
- האם הצומת השמאלי ביותר הוא תמיד עלה?
- היכן נמצא הערך הגדול ביותר בעץ?

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

הכנסת ערכים לעץ-חיפוש-בינרי ובניית עץ



הכנס 18

הכנס 7

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

שאלה

? ממשו את הפעולה :

public static void InsertIntoBST (BinTreeNode<**int**> bst, **int** x)

הפעולה מכניסה מספר שלם לעץ-חיפוש-בינרי המכיל מספרים

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

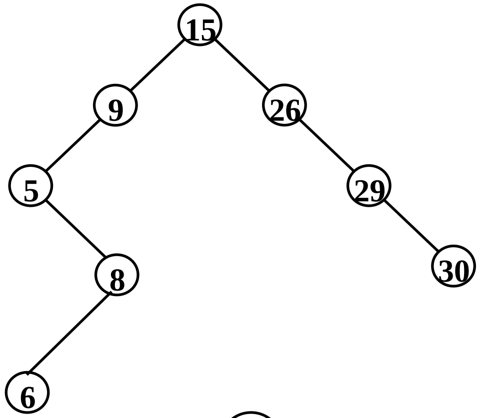
הכנסת ערכים לעץ-חיפוש-בינרי ובניית עץ

בעזרת פעולת ההכנסה שלמדנו ניתן ליצור עץ-חיפוש-בינרי:

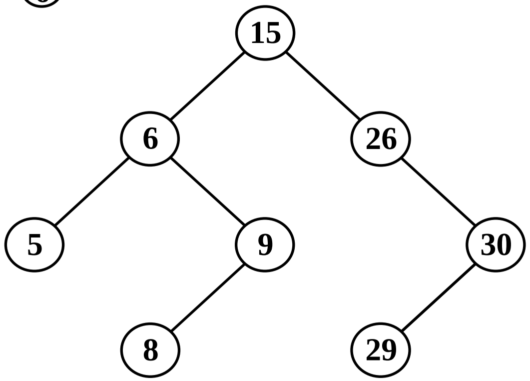
- ניצור עץ עלה
- נכניס אליו את הערכים באופן שלמדנו.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

בניית עץ



- 15, 9, 26, 5, 8, 6, 29, 30



- 15, 6, 5, 26, 30, 9, 29, 8

אותם איברים בסדר שונה,
גורמים לבניית עץ שונה

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

שאלה

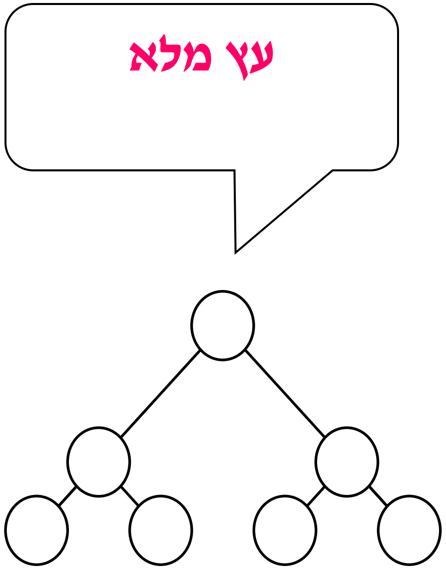
- ציירו עץ עבור כל אחת משתי סדרות האיברים הבאות:

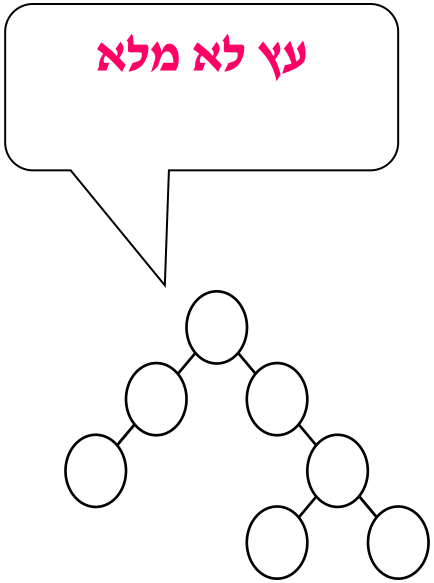
1. 30, 29, 26, 15, 9, 8, 6, 5
2. 26, 30, 15, 5, 29, 8, 9, 6

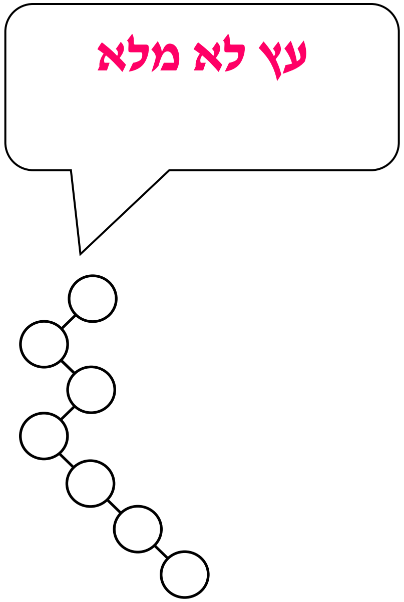
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

עץ מלא

עץ בינרי המלא בכל רמותיו נקרא עץ מלא:







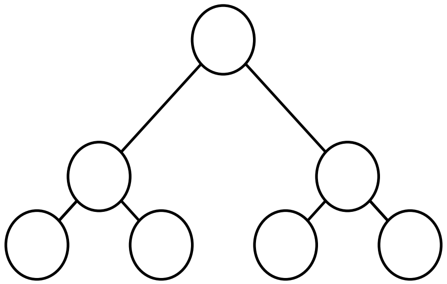
© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

יעילות הפעולות על עץ-חיפוש-בינרי

- ננתח את הפעולות בהנחה שהעץ מלא (או קרוב למלא).
- הפעולות שראינו על עץ-חיפוש-בינרי (חיפוש איבר, הכנסה) אינן עוברות על כל הצמתים בעץ.
- למעשה הפעולות עוברות על צומת אחד בלבד בכל רמה.
- לכן יעילות הפעולות תלויה בגובה העץ.
- מה הוא גובה העץ כאשר יש בעץ n צמתים?

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

גובה עץ



מה הוא גובה העץ בעל n צמתים?

רמה 0:	$2^0 = 1$
רמה 1:	$2^1 = 2$
רמה 2:	$2^2 = 4$
:	
רמה k:	2^k

בעץ בעל k רמות יש 2^k צמתים ברמה האחרונה. בכל שאר הרמות (בהנחה שהעץ הוא עץ מלא יש $2^k - 1$ צמתים. סה״כ לצורך ניתוח היעילות ניתן להגיד שיש 2^k צמתים.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

גובה עץ

אם יש 2^k צמתים,אזי נפתור את המשוואה :

$$2^k = n$$

נוציא log משני האגפים :

$$\log_2 2^k = \log_2 n$$

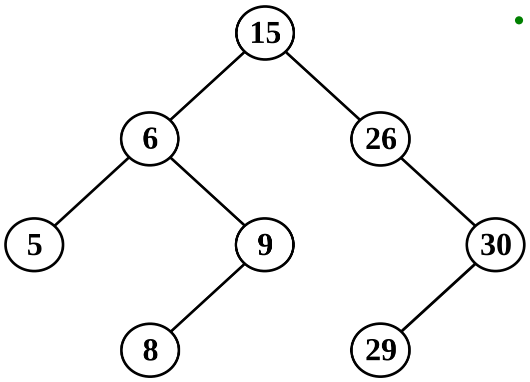
$$k \cdot \log_2 2 = \log_2 n$$

$$k = \log_2 n$$

ניתן להתעלם מהבסיס של הlogn (כי זו גם הכפלה בקבוע) ולטעון שמספר הרמות הוא O(logn)

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

מיון בעזרת עץ-חיפוש-בינרי



שאלה:

סרקו את עץ-החיפוש-הבינרי בסדר תוֹכי:

האם תמיד סריקה בסדר תוכי

מחזירה סדרה ממוינת? נמקו.

כתבו פעולה המקבלת עץ-חיפוש-בינרי ומדפיסה את איבריו בצורה ממוינת בסדר יורד.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

יעילות המיון בעזרת עץ-חיפוש-בינרי

- בניית עץ – לכל צומת מבצעים הכנסה : $O(n \log n)$
- סריקה תוֹכית – $O(n)$

יעילות המיון : $O(n \log n)$

מיון כזה נקרא מיון-עץ

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

מבני נתונים לעומת טיפוסי נתונים מופשטים

- מבני נתונים : שרשרת חוליות, עץ בינרי.
- טיפוסי נתונים מופשטים כלליים : מחסנית, תור.
- טיפוס נתונים מופשט שאינו כללי : רשימת תלמידים.
- טיפוס נתונים לא מופשט : רשימה.

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים

סוף

© המרכז להוראת המדעים
האוניברסיטה העברית בירושלים