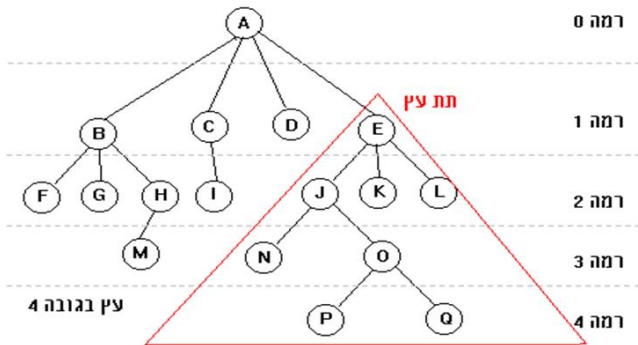


BinTreeNode - עצים בינאריים

עץ - מבנה נתונים (גרף) המורכב מצמתים המקיימים ביניהם יחס של הורה ובנים / צאצאים.

צומת - חוליה ולה תכונות: value - הפניה לטיפוס המידע, והפניות לצמתים אחרים.

קשת - הפניה היוצאת מצומת **אב** לצומת **בן**.



כל עץ מורכב מצומת **עלה** (root) שהוא הצומת הראשון בעץ.

לכל צומת פרט לשורש יש **הורה** / **אב** (parent).

כל צומת יכול להיות הורה לצמתים רבים אחרים,

וכל הצמתים הללו הם **בניו** של ההורה.

כל הצמתים שהם בניו של אותו הורה הם צמתים **אחיות**.

כל צומת בעץ הוא **צאצא** (descendent) של כל הצמתים שמעליו.

כל צומת הוא **אב קדמון** של כל צאצאיו.

מספר בניו של הצומת קובעים את דרגתו - **דרגת צומת** (degree).

לצומת שלו 3 בנים נקבעת דרגה 3 ואילו לצומת שאין לו בנים כלל נקבעת דרגה 0.

צומת שדרגתו 0 קרוי **עלה** (leaf).

עץ בינארי (BinNode) - עץ שבו הדרגה המקסימאלית של כל אחד מצמתיו הוא 2. כלומר - לכל צומת יש שתי

הפניות left ו-right לשני בנים

פילון - רצף של צמתים/קשתות היוצאות מצומת אחת עד צומת שני.

רמת עץ (tree level), אורך המסלול המקסימלי משורש העץ ועד העלה. השורש בעץ מסומן כרמה 0, בניו של

השורש נמצאים ברמה מספר 1, בני בניו ברמה 2 וכך הלאה.

רמה של צומת הינה מספר הקשתות שיש לעבור כדי להגיע משורש העץ עד הצומת המבוקש.

גובה עץ או עומק העץ הוא מספרה של הרמה הגבוהה ביותר בעץ.

מספר הקשתות לאורכו של המסלול הארוך ביותר היוצא מן השורש ומסתיים בעלה.

בן-אח (תת-עץ-שמאלי) - עץ בינארי ששורשו הוא הבן השמאלי של צומת נתון.

בן-אח (תת-עץ-ימני) - עץ בינארי ששורשו הוא הבן הימני של צומת נתון.

בן הוא תמיד שורש של **תת-עץ** (sub-tree).

רמה מלאה - רמה בעץ שקיימים בה כל הצמתים.

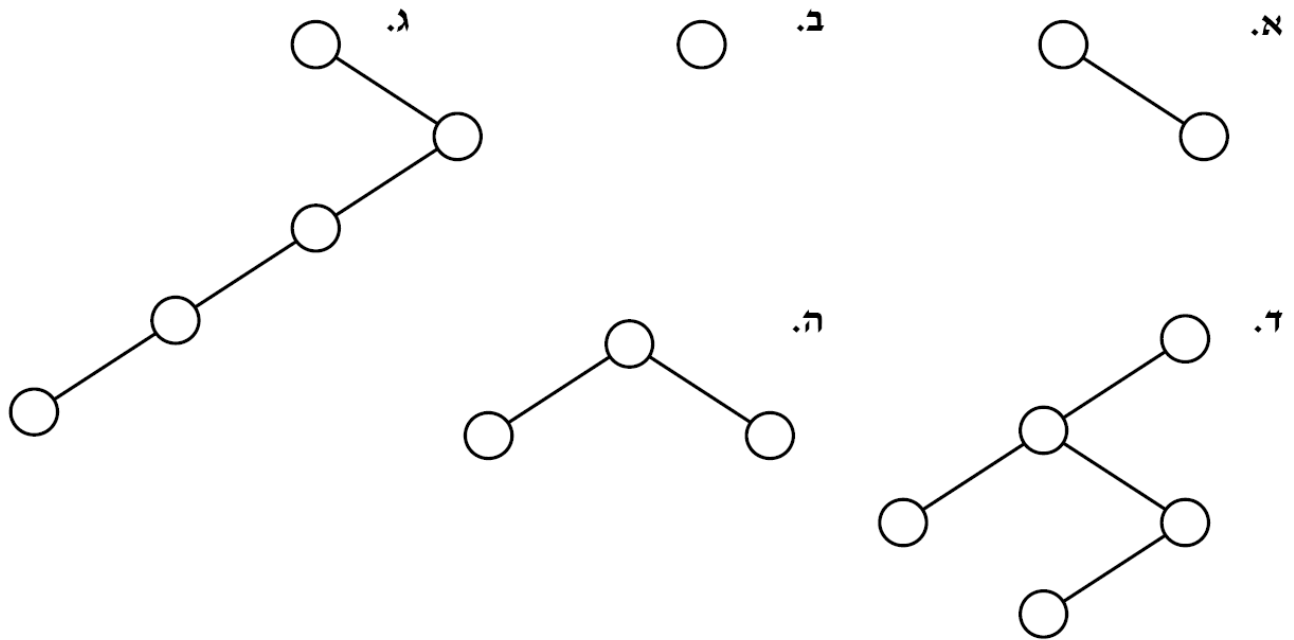
עץ בינארי מלא (עץ שלם) - עץ שכל רמה בו היא רמה-מלאה.

עץ בינארי כמעט מלא - עץ בינארי שכל הרמות פרט לרמה האחרונה מלאות, וכל הצמתים ברמה האחרונה מרוכזים בצד שמאל.

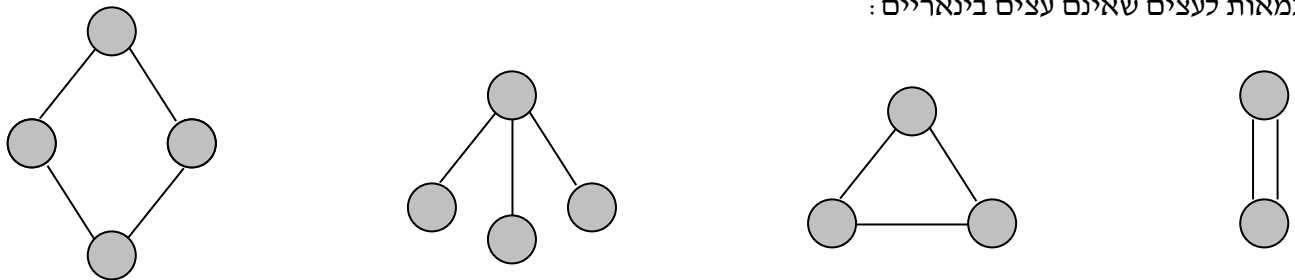
עץ בינארי לחלוטין (עץ לא בהכרח מלא) - עץ שדרגת כל צומת בו היא 0 או 2

(כלומר - אין צומת שלו בן יחיד).

דוגמאות לעצים בינאריים :

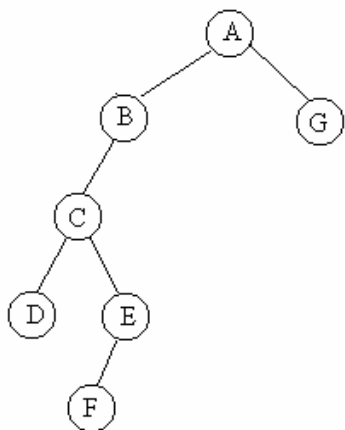


דוגמאות לעצים שאינם עצים בינאריים :



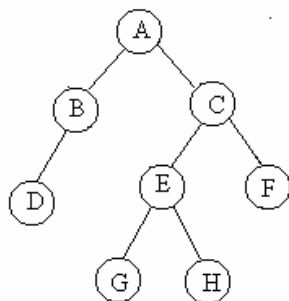
תרגילים:

1. אלו מהמשפטים הבאים נכונים לגבי העץ שלהלן? הסבר:



- א. A הוא אב קדמון של כל צמתי העץ האחרים. -----
- ב. B הוא אב קדמון של G. -----
- ג. F הוא צאצא של E, C ו-G. -----
- ד. D הוא ברמה 2. -----
- ה. D הוא אח של E. -----
- ו. C הוא אב קדמון של F ו-D. -----
- ז. רמתם של אחים בעץ זהה. -----

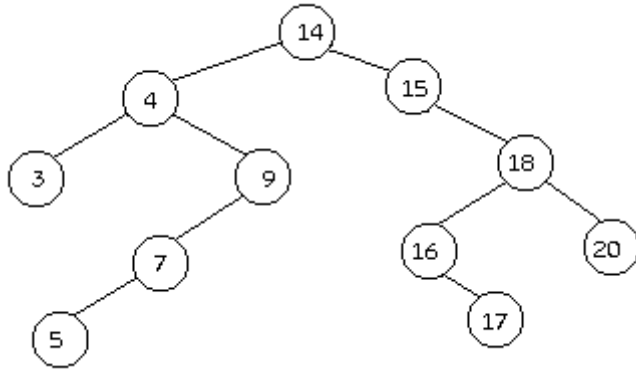
2. כמה אבות קדמונים יש לצומת ברמה n בעץ בינארי?



3. מהו מספר הצמתים ברמה n בעץ בינארי מלא?

4. מהו מספר הצמתים הכולל בעץ בינארי מלא שרמתו n?

סריקת עצים בינאריים:



pre order - סריקה בסדר תחילי

1. בקר בשורש.
2. סרוק תת-עץ השמאלי בסדר תחילי.
3. סרוק תת-עץ ימני בסדר תחילי.

in order - סריקה בסדר תוכי

1. סרוק תת-עץ השמאלי בסדר תוכי.
2. בקר בשורש.
3. סרוק תת-עץ ימני בסדר תוכי.

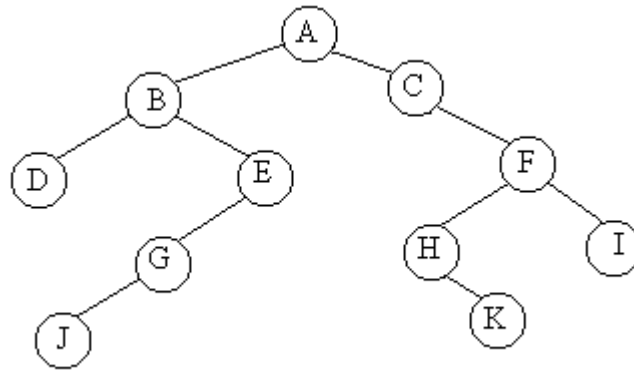
post order - סריקה בסדר סופי

1. סרוק תת-עץ השמאלי בסדר סופי.
2. סרוק תת-עץ ימני בסדר סופי.
3. בקר בשורש.

שלושת שיטות הסריקה מעידות על מיקומה של פעולת הביקור בשורש בין 3 הפעולות.

קיימת שיטת סריקה נוספת - סריקה לפי רמות.

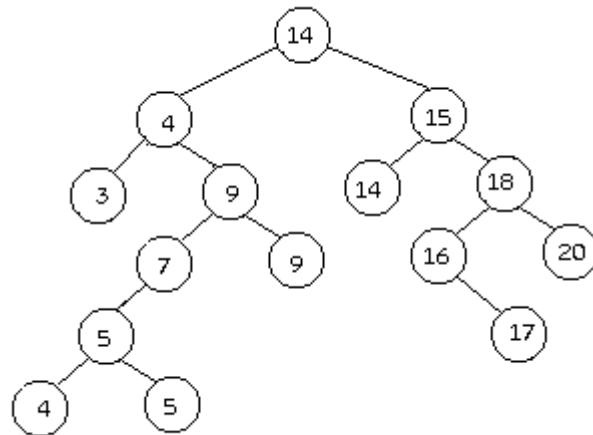
- שחזור עצם בינארי על פי 2 מסריקותיו.



סריקת העץ בשיטות השונות:

PreOrder	A B D E G J C F H K I	: סריקה בסדר תחילי:
InOrder	D B J G E A C H K F I	: סריקה בסדר תוכי:
PostOrder	D J G E B K H I F C A	: סריקה בסדר סופי:

מהו סדר הביקור בצמתים בכל אחת משלוש שיטות הסריקה? **תרגיל:**



in order: _____

pre order: _____

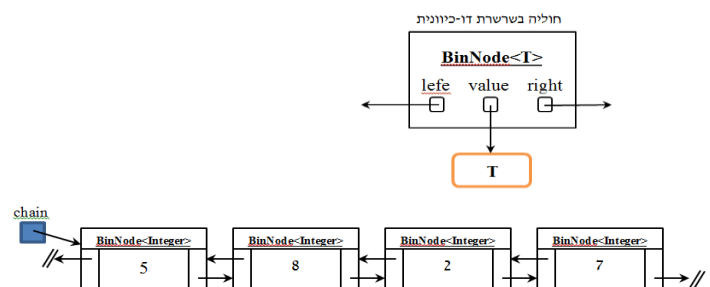
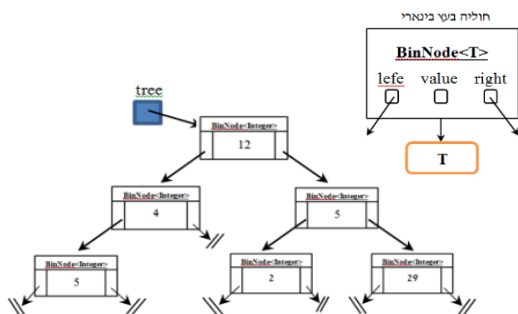
post order: _____



חוליה בינארית <T> BinNode

חוליה המכילה ערך מטיפוס T והפנייה לשתי חוליות בינאריות. ניתן להשתמש במחלקה זאת כדי לייצג עץ בינארי או שרשרת דו-כיוונית.

סיבוכיות	חתימת הפעולה	תיאור הפעולה
		בנאי:
O(1)	BinNode (T x)	פעולה הבונה חוליה בינארית שערכה x ושתי ההפניות null
O(1)	BinNode (BinNode<T> left, T x, BinNode<T> right)	פעולה הבונה חוליה בינארית שערכה x, ושתי ההפניות שלה הן left ו- right (ערכם של הפרמטרים left ו- right יכול להיות null)
		שאלות:
O(1)	T getValue ()	אם T מחלקה עוטפת לטיפוס בסיסי (Integer, Double, Character) יוחזר ערך החוליה, ואם הפניה לעצם, תוחזר הפנייה לעצם זה
O(1)	BinNode <T> getLeft ()	פעולה המחזירה הפנייה לחוליה אליה מפנה left.
O(1)	BinNode <T> getRight ()	פעולה המחזירה הפנייה לחוליה אליה מפנה right.
O(1)	boolean hasLeft ()	פעולה המחזירה אמת אם יש חוליה משמאל, ושקר אחרת
O(1)	boolean hasRight ()	פעולה המחזירה אמת אם יש חוליה מימין, ושקר אחרת
O(T)	String toString ()	פעולה המחזירה את מצב החוליה כמחרוזת. (*) סיבוכיות: אם T עצם מטיפוס פשוט $O(1) \leftarrow$ ואם T מייצג אוסף באורך כלשהו $O(T) \leftarrow$
		פקודות:
O(1)	void setValue (T x)	הפעולה משנה (מעדכנת) את ערך החוליה ל- x
O(1)	void setLeft (BinNode <T> left)	הפעולה משנה את ערכה של ההפניה left ל- left הפרמטר left יכול להיות גם null
O(1)	void setRight (BinNode <T> right)	הפעולה משנה את ערכה של ההפניה right ל- right הפרמטר right יכול להיות גם null



תכנית לבניית עצם בין אריות

עקבו אחר קוד המחלקה וציירו את העץ המתקבל:

```
import unit4. collectionsLib.*; // חתייחס ל-unit4 החדש
import unit4. utilsLib.*;

public class PrintTree
{
    public static void main(String[] args)
    {
        BinNode<Character> t1, t2, bt1, bt2, bt;

        //--- ..... ---
        t1 = new BinNode <Character>('K');
        t2 = new BinNode <Character>('M');
        t2.setLeft(new BinNode <Character>
            (null, 'F', new BinNode <Character>('L')));

        bt1 = new BinNode <Character>(t1, 'B', t2);
        bt1 = new BinNode <Character>(null, 'J', bt1);

        //--- ..... ---
        t1 = new BinNode <Character>('D');
        t2 = new BinNode <Character>('C');

        t2.setLeft(new BinNode <Character>('I'));
        t2.setRight(new BinNode <Character>
            (new BinNode <Character>('G'), 'H', null));

        bt2 = new BinNode <Character>(t1, 'E', t2);
        bt = new BinNode <Character>(bt1, 'A', bt2);

        TreeUtils.showTree(bt, "bt");
    }
}
```

- א. צייר את העץ המתקבל
ב. העתק את התכנית למחשב ובדוק את נכונות השרטוט שהתקבל בסעיף א'.

תכנית לבניית עצץ בינארי

העץ המתקבל:

```
import unit4. collectionsLib.*;
import unit4. utilsLib.*;

public class PrintTree
{
    public static void main(String[] args)
    {
        BinNode<Character> t1, t2, bt1, bt2, bt;

        //--- בניית ענף שמאלי ---
        t1 = new BinNode<Character>('K');
        t2 = new BinNode<Character>('M');
        t2.setLeft(new BinNode<Character>
            (null, 'F', new BinNode<Character>('L')));

        bt1 = new BinNode<Character>(t1, 'B', t2);
        bt1 = new BinNode<Character>(null, 'J', bt1);

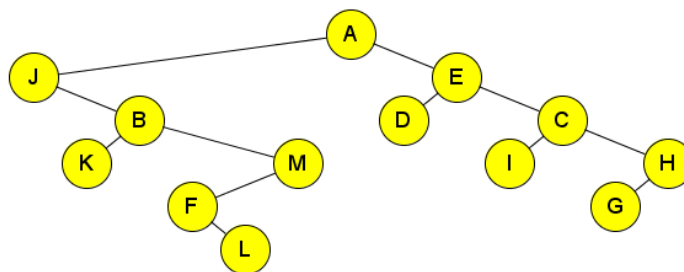
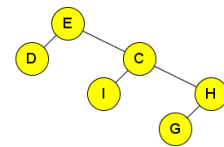
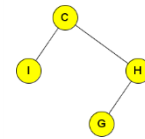
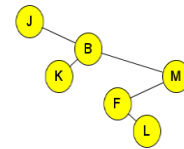
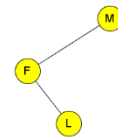
        //--- בניית ענף ימני ---
        t1 = new BinNode<Character>('D');

        t2 = new BinNode<Character>('C');
        t2.setLeft(new BinNode<Character>('I'));
        t2.setRight(new BinNode<Character>
            (new BinNode<Character>('G'), 'H', null));

        bt2 = new BinNode<Character>(t1, 'E', t2);

        bt = new BinNode<Character>(bt1, 'A', bt2);

        TreeUtils.showTree(bt, "bt");
    }
}
```



אוסף פגולות לצד ה'ארייט:

- מספר הצמתים בעץ
- סכום הצמתים בעץ
- עלה?
- מספר העלים בעץ
- סכום העלים
- מספר הצמתים שערכם x
- מספר צמתים זוגיים
- מספר בנים ימניים בעץ
- מספר בנים יחידים בעץ
- מספר הבנים של צומת נתון
- מספר הנכדים של צומת נתון
- עץ פרו ורבו (2004) - מחזיר אמת אם קיים צומת ולו לפחות שני נכדים משני הבנים.
- מספר-סבים בעץ
- האם-נמצא? (bt, x)
- האם כל הצמתים אי-זוגיים?
- עץ-ממוצע? - אמת אם ערך כל צומת שווה למוצע הערכים של בניו.
- ממוין-רמות? - אמת אם ערך כל אב קטן מערך בניו, ושקר אחרת
- סריקה לפי רמות
- רמת-צומת (bt, t) - t הפניה לצומת בעץ bt
- סכום צמתים ברמה $(bt, level)$
- מספר צמתים ברמה $(bt, level)$
- האם אחים? $(bt, t1, t2)$
- גובה-עץ
- ערך מקסימאלי בעץ
- סכום צמתים ברמה
- מספר צמתים ברמה
- האם-אחים? $(bt1, bt2)$
- האם-הורה? $(bt1, bt2)$

- סבא רבא (bt, tX, tY) - בגרות - מחזיר הפניה לצומת שהוא אב-קדמון הכי קרוב לשני עלים נתונים.
- צאצא? $(t1, t2)$ מחזיר אמת אם הצומת $t1$ הוא צאצא של $t2$ ושקר אחרת
- הורה (bt, son) - מחזיר הפניה לצומת ההורה של הצומת son . הנחה: $bt \neq son$
- מסלול-אחיד - האם קיים מסלול מצומת השורש לאחד העלים שערכי כל צמתיו זהה.
- מספר צמתים ברמה k
- עץ-מלא? - כל הרמות מלאות (הקשר שבין מספר הצמתים N וגובה העץ K : $2^k = n$)
- עץ בינארי לחלוטין (עץ מלא) - עץ שדרגת כל צומת בו היא 0 או 2
- בגרות 2007 - מסלול אחיד (בגרות 1996 - עץ בעל שרשרת x)
- עץ ביטוי
- עץ חיפוש בינארי
- עץ ממומש במערך (1999)
- עץ שה- $value$ שלו הוא מבנה נתונים מסוג: מערך / רשימה / מחסנית / תור / עץ
- ייצוג שונה לעץ בינארי (באמצעות מערך)
- עץ טרינארי