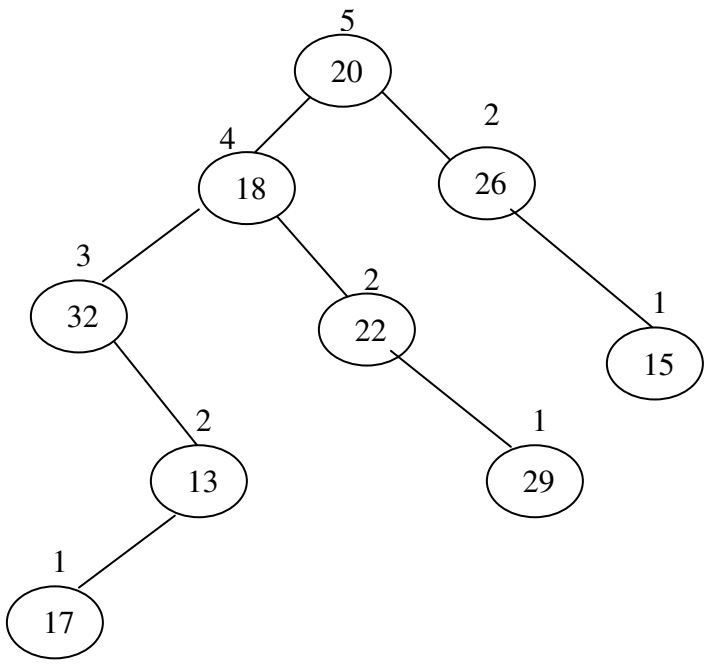
להלן תוכנית בשם move-one-right(T,x) שמקבלת קלט עץ בינארי עם גבהים T וצומת x בעץ ומחזירה את הצומת שנמצאת מימין לצומת x באותה הרמה של צומת x. אם אין צומת כזאת התוכנית מחזירה NULL.

<u>דוגמה:</u>
יהי T עץ בינארי עם גבהים שמתואר בציור הבא : מעל כל צומת מצוין הגובה שלו.



לאחר הקריאה לפונקציה move-one-right(T,13) הפונקציה תחזיר את צומת 29
לאחר הקריאה לפונקציה move-one-right(T,22) הפונקציה תחזיר את צומת 15
לאחר הקריאה לפונקציה move-one-right(T,29) הפונקציה תחזיר NULL

<u>הרעיון:</u>

נטפס למעלה בעץ החל מצומת x, נסמן ב- z את הצומת שהגענו אליה במהלך הטיפוס למעלה ונסמן ב- p את האבא של z. נשמור במשתנה t את מספר הצעדים שעלינו בטיפוס למעלה.
אם z בן ימני של אביו p נבצע z=p , t++ ונמשיך לטפס למעלה.
אם z בן שמאלי של אביו p יש שתי אפשרויות:
1) אם גובה הבן הימני של p הוא כזה שאפשר לרדת בו למטה ולהגיע לצומת ברמה של x אז נעבור לבן הימני של p ונבצע ירידה למטה עד שנגיע לצומת הרצוי.
2) אם גובה הבן הימני של p קטן מדי ולא מאפשר לרדת למטה את אותו מספר צעדים שעלינו למעלה מ- x עד ל- p נבצע z=p, t++ ונמשיך לטפס למעלה.
אם במהלך הטיפוס למעלה הגענו לכך שצומת z הוא השורש זה סימן ש- x היה הימני ביותר ברמה שלו ונחזיר NULL.

להלן הפסיאודו קוד של התוכנית:

move-one-right(T,x)

```
t=1
z=x
while (parent(z)≠ NULL) {
  p=parent(z)
  if (z=right(p)){
     t++
     z=p
     continue
     }
  if z=left(p) {
     if (right(p)≠NULL)&& (h(right(p)≥ t){
        return leftmost-in-level(T_right(p),t)
        }
     t++
     z=p
     }
  }
return NULL
```

התוכנית הנ"ל משתמשת בתוכנית עזר בשם leftmost-in-level(T,level) שמקבלת כפרמטרים עץ עם גבהים T ורמה level ומחזירה את הצומת השמאלי ביותר ברמה level בעץ.
התוכנית מניחה שיש לפחות צומת אחד ברמה level בעץ.
לדוגמה הקריאה לתוכנית: leftmost-in-level(T,4) עבור העץ T שבציור בעמוד הקודם מחזירה את הצומת 13

להלן הפסיאודו-קוד של תוכנית העזר:

leftmost-in-level(T,level)

```
x=root(T)
t=level
while (t>1) {
  if (left(x)≠ NULL) && (h(left(x))≥ t-1){
      t--
      x=left(x)
      continue
  }
  t--
  x=right(x)
}
return x
```