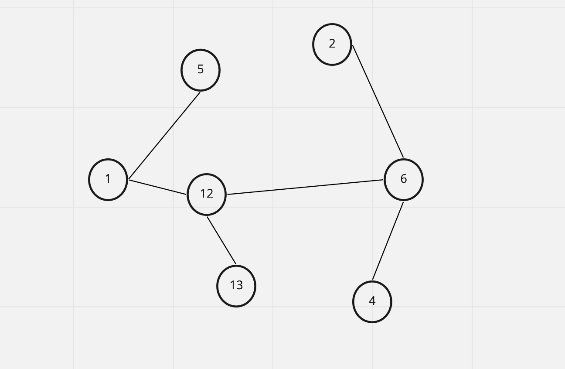
אלגוריתם שריפת עלים  
מטרה: בהיתן גרף T (הנתון כרשימת שכנויות) המייצג עץ (גרף קשיר ללא מעגלים) יש למצוא את המרכז של העץ.  
מרכז = הקודקוד שהמרחק הכי גדול ממנו לקודקוד אחר הוא מינימאלי.



בציור, קודקוד המרכז הוא 12 כיוון שהמרחק הכי גדול שלו מכל קודקוד אחר הוא 2 שזה הכי קטן ביחס לאחרים.  
מרחק הכי גדול נקרא אקסצנטריות. בציור:

ברגע שמצאנו מרכז :   
הרדיוס הוא ה .  
הקוטר הוא המרחק הכי גדול בין 2 קודקודים בגרף וניתן לחשבו ע"י: .

בעץ יכול להיות או מרכז אחד או 2 מרכזים (בגרף כללי יכולים להיות הרבה מרכזים).  
אם יש 2 מרכזים . הרדיוס יהיה .  
הקוטר:

אלגוריתם:

הרעיון הוא: להוריד מהגרף בכל שלב את העלים ולקבל גרף קטן יותר עם עלים חדשים ובכך "לסגור" על האמצע.

בהינתן גרף G עם n קודקודים.

* אתחל count=n (מספר הקודקודים שנשארו בגרף בכל שלב)
* אתחל radius=0
* אתחל מערך דרגות השומר את כל הדרגות של הקודקודים בגרף.
* אתחל רשימה ריקה L והוסף אליה את כל העלים בגרף G (מי שה שלו שווה 1)
* אתחל רשימת עזר . (רשימת העלים החדשים לאיטרציה הבאה)
* כל עוד (לא נשארו רק המרכזים אלא יש עוד עלים להוריד)
  + הגדר: (שריפת כל העלים ב L מקדמת אותנו בצעד אחד למרכז)
  + עבור כל עלה u ב L
    - מחק את u מ L והגדר: .
    - הגדר:
    - הורד את הדרגה של השכן v של u. כלומר: .
    - אם דרגת השכן v של u היא 1 הוסף את v ל (אי אפשר להוסיף אותו כרגע ל L כי היא עדיין לא ריקה מהאיטרציה הנוכחית)
  + נקה את

סיבוכיות: אתחול מערך דרגות ורשימת עלים - כאשר הוא מספר הקודקודים בגרף.  
אתחול , radius, count ורשימת עזר ריקה -

נשים לב שסה"כ בכל הלולאות, כל קודקוד ייכנס ל L בדיוק פעם אחת (כי לאחר מכן דרגתו תהיה 0 והוא לא יחשב כחלק מהקודקודים בגרף) בנוסף, חיפוש קודקוד שכן "פעיל" (שלא נמחק לפני כן) בתוך קודקוד עלה שעכשיו נמחק תתבצע רק פעם אחת עובר כל קודקוד ולכן סה"כ פעולה זו תתבצע: פעמים.  
כל שאר הפעולות הפנימיות ייתבצעו פעם אחת לכל קודקוד ולכן סה"כ: פעמיים.

לכן סיבוכיות האלגוריתם כולו היא: .

קוד:

public class Fire {

int center1, center2, radius, diameter; // תשובות

public Fire(ArrayList<Integer>[] graph) {

int n = graph.length; // |V|

int[] deg = new int[n]; // מערך דרגות

ArrayList<Integer> leaves = new ArrayList(); // L

ArrayList<Integer> newLeaves = new ArrayList(); // L’

int r = 0; // רדיוס

for (int i = 0; i < n; i++) {

deg[i] = graph[i].size(); // אתחול הדרגה למספר השכנים

if (deg[i] == 1) leaves.add(i);

}

while (n > 2) {

newLeaves = new ArrayList();

r++;

while (!leaves.isEmpty()) {

int u = leaves.remove(0);

deg[u]=0;

n--;

for(int v : graph[u]) { // עבור כל ערך בתוך הרשימה -השכנים

if(deg[v] != 0) {

deg[v]--;

if (deg[v] == 1) { newLeaves.add(v); }

}

}

}

leaves = newLeaves;

}

if (n == 2) {

center1 = leaves.remove(0);

center2 = leaves.remove(0);

radius = r+1;

diameter = 2\*r+1;

} else {

center1 = center2 = leaves.remove(0);

radius = r;

diameter = 2\*r;

}

}

}

הוכחה:   
טענה: בסיום האלגוריתם, ברשימה L יהיו המרכזים. ומספר האיטרציות של הלולאה החיצונה הוא הרדיוס.  
האלגוריתם מסתיים כי לפי משפט: "בכל עץ יש לפחות עלה אחד" והורדת כל העלים מעץ מייצרת עץ.  
לכן בכל שלב יהיו עלים חדשים ולכן count ירד לפחות ב 1 בכל איטרציה והאלגוריתם יעצור.  
טענה 1: אם מרכז יחיד, הוא מספר האיטרציות של הלולאה החיצונה אז: וכן L מכילה את c.

הוכחה: באינדוקציה על ה .  
בסיס: אז העץ מכיל רק את c ולכן הלולאה החיצונה לא תתבצע בכלל (כי ) ולכן: .  
צעד: נניח שלכל גרף המקיים: אז: . נוכיח שהטענה נכונה עבור n+1.  
יהא T עץ עם מרכז c המקיים: . מכאן: . (כי אם היו רק 2 קודקודים אז המרחק בין שניהם היה 1 ושניהם היו מרכזים אבל נתון שיש רק מרכז אחד). לכן לפחות איטרציה אחת תתבצע.  
באיטרציה זו, מורידים את כל העלים בעץ T ומקבלים עץ חדש (לפי הטענה שהורדת עלים מעץ משאירה עץ) . המקיים: לכל קודקוד בגרף . כי מסלול ארוך ביותר מקודקוד בעץ (שאינו עלה) מסתיים בהכרח בעלה (אחרת ניתן להאריכו) ולכן הורדת העלים מ T קיצרה את כל המסלולים (שמתחילים מקודקוד פנימי) ב 1. אבל מכאן: מכיוון שכולם ירדו ב 1 אז המינימאלי ב T נשאר המינימאלי ב . לכן: הוא גם מרכז . לכן: . לכן לפי הנחת האינדוקציה, האלגוריתם יבצע עוד n איטרציות החל מ ובסה"כ יהיו איטרציות.  
מכיוון שראינו כי נשאר המרכז בכל איטרציה נקבל שבסוף נישאר רק עם c. באיטרציה לפני האחרונה, c יהפוך להיות עלה כאשר נמחק את הקודקוד האחרון פרט אליו ולכן c ייכנס ל L.  
  
טענה 2: אם יש 2 מרכזים, הוא מספר האיטרציות של הלולאה החיצונה אז: וכן L מכילה את .

הוכחה: באינדוקציה על ה .

בסיס: אז העץ מכיל רק את ולכן הלולאה החיצונה לא תתבצע בכלל (כי ) ולכן: .  
צעד: נניח שלכל גרף המקיים: אז: . נוכיח שהטענה נכונה עבור n+1.  
יהא T עץ עם מרכזים המקיים: . מכאן: . לכן לפחות איטרציה אחת תתבצע.  
באיטרציה זו, מורידים את כל העלים בעץ T ומקבלים עץ חדש (לפי הטענה שהורדת עלים מעץ משאירה עץ) . המקיים: לכל קודקוד בגרף . כי מסלול ארוך ביותר מקודקוד בעץ (שאינו עלה) מסתיים בהכרח בעלה (אחרת ניתן להאריכו) ולכן הורדת העלים מ T קיצרה את כל המסלולים (שמתחילים מקודקוד פנימי) ב 1. אבל מכאן: מכיוון שכולם ירדו ב 1 אז המינימאלי ב T נשאר המינימאלי ב . לכן: הם גם מרכזים ב . לכן: . לכן לפי הנחת האינדוקציה, האלגוריתם יבצע עוד n איטרציות החל מ ובסה"כ יהיו איטרציות.  
מכיוון שראינו כי נשארו המרכזים בכל איטרציה נקבל שבסוף נישאר רק עם . באיטרציה לפני האחרונה, יהפכו להיות עלה (כי לא ייתכן שרק אחד מהם יהפוך להיות עלה לפני השני כיוון שהם המרכזים בכל איטרציה ואם בה"כ הוא עלה אז הוא מחובר רק ל כיוון ש 2 המרכזים מחוברים ביניהם(\*) ואז המרחק של יהיה גדול ב 1 מהמרחק של משאר הקודקודים וזה לא ייתכן כי שניהם מרכזים). כאשר נמחק את הקודקודים האחרונים פרט אליהם אז ייכנסו ל L.

הוכחה ששני מרכזים בעץ מחוברים ביניהם: נניח בשלילה שלא ולכן קיים במסלול בין ל .  
(יש מסלול בין כל 2 קודקודים כי בגרף קשיר).  
כי u הוא לא מרכז. יהי הקודקוד הכי רחוק מ ולכן הוא עלה.  
מכאן אורך המסלול בין ל הוא ומכאן: הכי רחוק מ .  
מכיוון ש עלה אז הוא לא על המסלול ביניהם: מכאן קיים מסלול מ ל .  
וקיים מסלול מ ל ולפחות אחד מהם עובר דרך u כי אחרת נקבל מעגל (לא פשוט) בין ולכן יש גם מעגל פשוט - סתירה.