אלגוריתם Floyd Warshall:

**מטרה**: בהינתן גרף G המיוצג ע"י מטריצת שכנויות, הגרף יכול להיות ממושקל (יש משקלים על הצלעות).

רוצים למצוא מסלולים קצרים ביותר (או קלים ביותר) בין כל 2 קודקודים בגרף.

**האלגוריתם**:  
1. נאתחל מטריצת תשובה A באופן הבא: אלכסון - 0. אם המשקל במטריצת השכנויות היה 0 נאתחל את אותו מיקום באינסוף. אם המשקל שונה מ 0 - נעתיק אותו גם למטריצת התשובה.  
2. עבור k מ 1 עד n   
 עבור i מ 1 עד n

עבור j מ 1 עד n

3. מחזירים את מטריצת A

**סיבוכיות**: אתחול מטריצת התשובה: .  
מילוי המטריצה: כי עבור כל קודקוד k אנו עוברים על כל זוג קודקודים i,j ומנסים למצוא מסלול קצר מ i ל j דרך k.

**הוכחה**: נסמן ב את אורך(משקל) המסלול הקצר ביותר בין קודקוד i ל j.  
נסמן ב את אורך(משקל) המסלול הקצר ביותר בין קודקוד i ל j שעובר בדרך רק בקודקודים: (פרט ל i,j עצמם)  
נוכיח באינדוקציה על k שלאחר איטרציה k של האלגוריתם: ומזה ינבע שלאחר איטרציה אחרונה נקבל: .  
בסיס: . נקבל מכאן שאסור לעבור דרך אף קודקוד באמצע אלא רק ישירות מ i ל j.  
לכן: וכן אם i,j לא מחוברים בצלע. אם i,j מחוברים בצלע שמשקלה c.  
השוויון בכל אחד מהמקרים ל נובע מהאתחול של האלגוריתם.

צעד: נניח שהטענה נכונה עד k כלשהו ונוכיח עבור k+1.  
לפי הנחת האינדוקציה: לכל i,j. צ"ל: לאחר איטרציה k+1 מתקיים: .  
נחלק למקרים:  
1. אם המסלול הקצר ביותר בין i ל j שעובר דרך הקודקודים: לא עובר דרך קודקוד k+1.  
אז: .  
2. אם המסלול הקצר ביותר בין i ל j שעובר דרך הקודקודים: כן עובר דרך הקודקוד k+1 אז נחלק את המסלול ל 2 מסלולים: מסלול מ i ל k+1 ומסלול מ k+1 ל j. שני המסלולים שהגדרנו כעת עוברים רק דרך קודקודים מהקבוצה: . לכן לפי הנחת האינדוקציה: ..   
המינימום מבין 2 המקרים הוא המסלול הקצר ביותר בין i ל j שעובר דרך .  
לפי האלגוריתם, באיטרציה ה k+1 בוחרים את המינימום מבין: ולפי הנחת האינדוקציה ולפי 1,2 מתקיים:

**פסאודו קוד**:

public static double[][] FW(int[][] graph) { // O(N^3)

int n = graph.length;

double[][] ans = new double[n][n];

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

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

if(i == j) ans[i][j] = 0;

else if(graph[i][j] == 0) ans[i][j] = Double.POSITIVE\_INFINITY;

else ans[i][j] = graph[i][j];

}

}

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

for (int i = 0; i < ans.length; i++) {

for (int j = 0; j < ans.length; j++) {

ans[i][j] = Math.min(ans[i][j], ans[i][k] + ans[k][j]);

}

}

}

return ans;

}