# גרפים

- בהינתן גרף $G = (V,E)$, נסמן: $n = |V|, m = |E|$

**הגדרות:**

גרף – זוג $G = (E,V)$ כאשר E קבוצת קשתות ו-V קבוצת קודקודים.

בגרף מכוון – $E \subseteq \{(u,v) | u,v \in V\}$

בגרף לא מכוון- $E \subseteq \{\{u,v\} | u,v \in V\}$

גרף ממושקל: $G = (E,V,W)$ כאשר $W: E \to \mathbb{R}$.

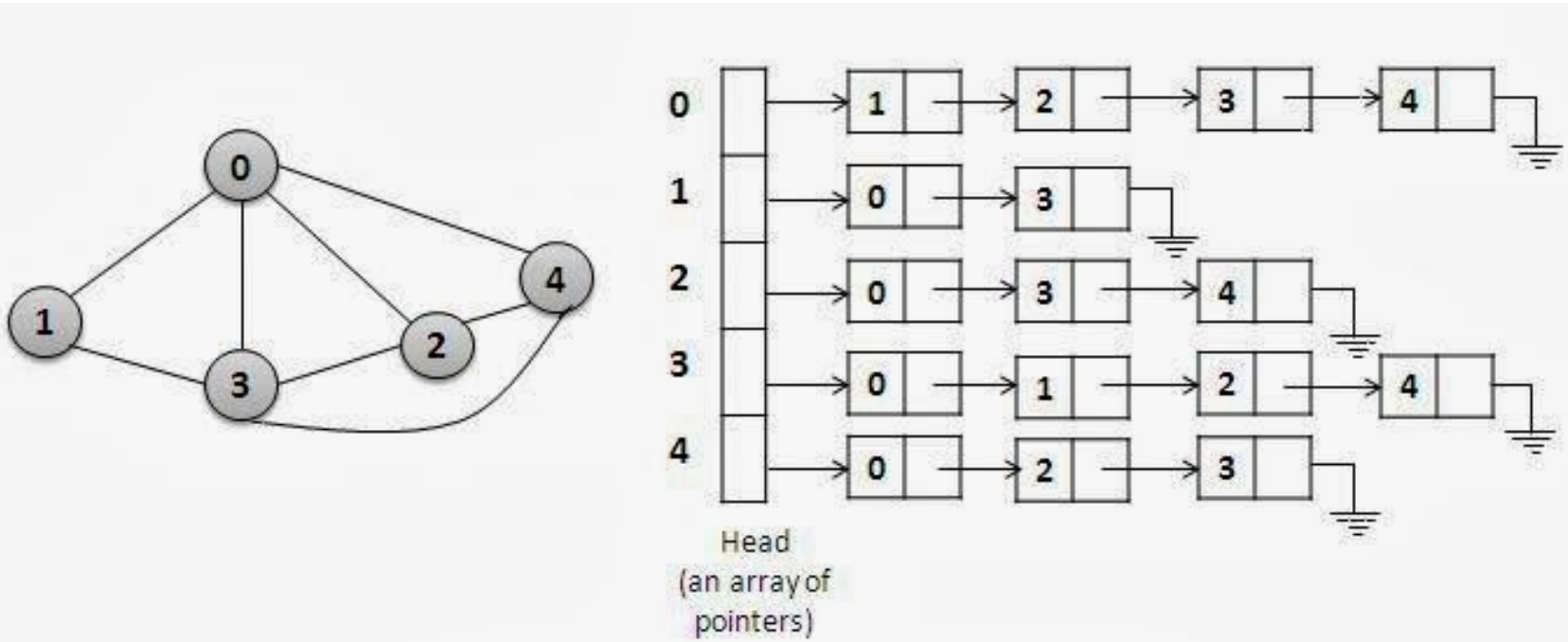
**ייצוג של גרפים:**

רשימת שכנויות – טוב לגרפים דלילים. מערך של קודקודים ולכל אחד יש מערך של מצביעים לקודקודים אליהם יש קשת.

מטריצת שכנויות – טוב לגרפים צפופים.

ברשימת שכנויות:

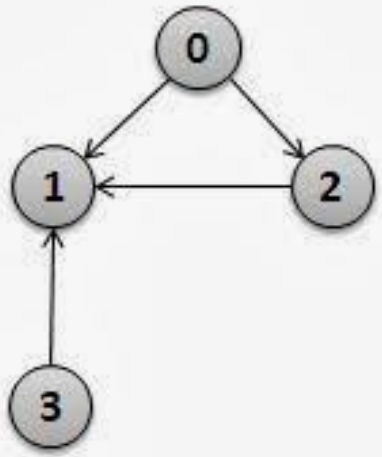
- בגרפ מכוון – סכום אורכי הרשימות הוא $|E|$. דוגמא – *(u,v)* מיוצג ברשימה של *u* בלבד.
- בגרפ לא מכוון – סכום אורי הרשימות הוא $2|E|$. לדוגמא – *(u,v)* מיוצג ברשימה של *u* וגם של *v*.
- בגרפ ממושקל – המשקל *w(u,v)* של קשת מאוחסן עם קודקוד *v* ברשימה של *u*.
- חיסרון – יש צורך לעבור על כל הרשימה כדי לחפש האם קשת קיימת.




**Adjacency List Representation of Graph**

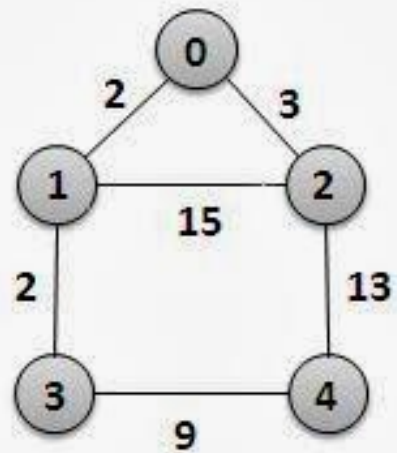
במטריצת שכנויות:

- הקודקודים ממוספרים בצורה שרירותית $1,2,\ldots,|V|$. המטריצה $A$ שמימדיה $|V|x|V|$ מייצגתאת גרף $G$ וערכיה הם: $a_{ij} = \begin{cases} 1 & (i,j) \in E \\ 0 & (i,j) \notin E \end{cases}$.
- נגדיר את $A^T$ להיות המטריצה המשוחלפת של $A$. בגרף לא מכוון $A = A^T$.
- בגרף לא מכוון – אפשר לשמור רק חצי מטריצה (חוסך מקום).
- ייצוג בגרף ממושקל: $a_{ij} = \begin{cases} w(i,j) & (i,j) \in E \\ 0 \; or \; \infty & (i,j) \notin E \end{cases}$.




| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 0 | 0 |

**Adjacency Matrix Representation of Directed Graph**



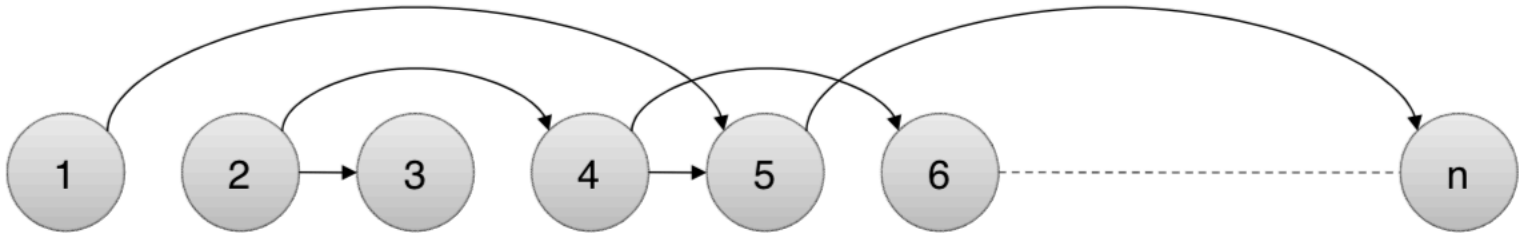

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 2 | 3 | 0 | 0 |
| 1 | 2 | 0 | 15 | 2 | 0 |
| 2 | 3 | 15 | 0 | 0 | 13 |
| 3 | 0 | 2 | 0 | 0 | 9 |
| 4 | 0 | 0 | 13 | 9 | 0 |

**Adjacency Matrix Representation of Weighted Graph**

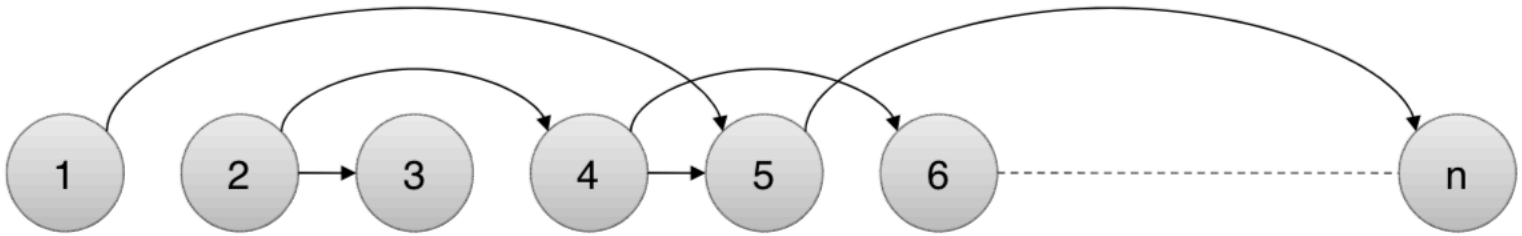
# תרגיל

נתון גרף מכוון $G = (V, E)$ שצמתיו ממוספרים: $1,2,\ldots,n$ $(n = |V|)$.

בנוסף, נתון שכל קשת $(i,j) \in E$ בגרף מקיימת $i < j$.




מצאו אלגוריתם יעיל המחשב את אורך המסלול הארוך ביותר בגרף.



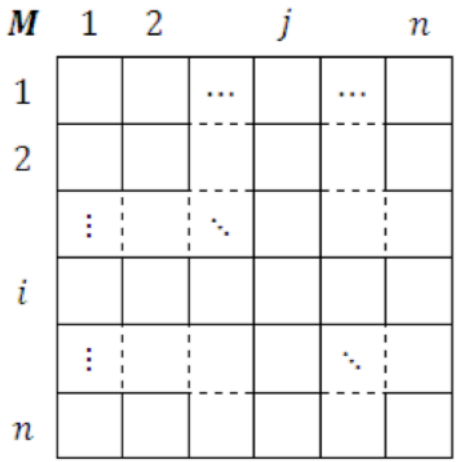

נחשב לכל צומת את אורך המסלול הארוך ביותר היוצא ממנו.
נחשב ערכים אלו במערך $L$ שאורכו $n$.
נשים לב כי $L[n] = 0$, כיוון שמצומת זה לא יכולות לצאת קשתות.
כנ"ל לכל צומת ללא קשתות יוצאות.

נעבור על הצמתים מהסוף להתחלה, החל מצומת $n-1$ עד לצומת $1$.
לכל צומת $i$ בעל שכנים $j_1, j_2, \ldots, j_k$, אורך המסלול המקסימלי היוצא ממנו הוא:

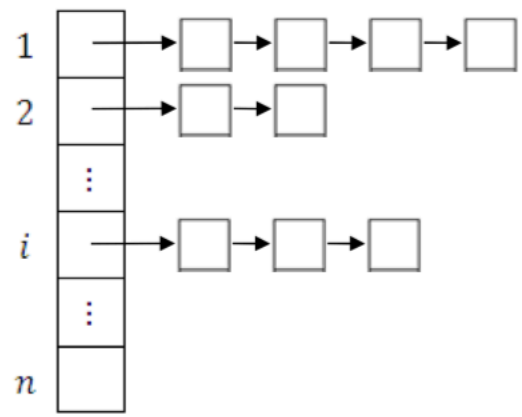
$$L[i] = 1 + \max\{\ L[j_1], L[j_2], \ldots, L[j_k]\ \}$$

היות ולכל שכן $j$ של צומת $i$ גדול ממנו, כבר חישבנו עבורו את $L[j]$.
לכל צומת $i$, זמן החישוב של $L[i]$ הוא כזמן אשר לוקח לעבור על שכניו.

מה סיבוכיות הזמן של האלגוריתם?




- אם הגרף מיוצג ע"י <u>מטריצת סמיכויות</u>, מעבר על שכני כל צומת בגרף ייקח $\Theta(n)$.
  לכן, סיבוכיות הזמן הכוללת של האלגוריתם היא $\Theta(n^2)$.




- אם הגרף מיוצג ע"י <u>רשימת סמיכויות</u>, מעבר על שכני כל צומת $i$ ייקח $\Theta(d_i)$, כאשר $d_i$ דרגת הצומת.
  לכן, סיבוכיות הזמן הכוללת של האלגוריתם היא:

$$\Theta\left(n + \sum_{i=1}^{n} d_i\right) = \Theta(n+m)$$

**חיפוש לרוחב: *BFS***

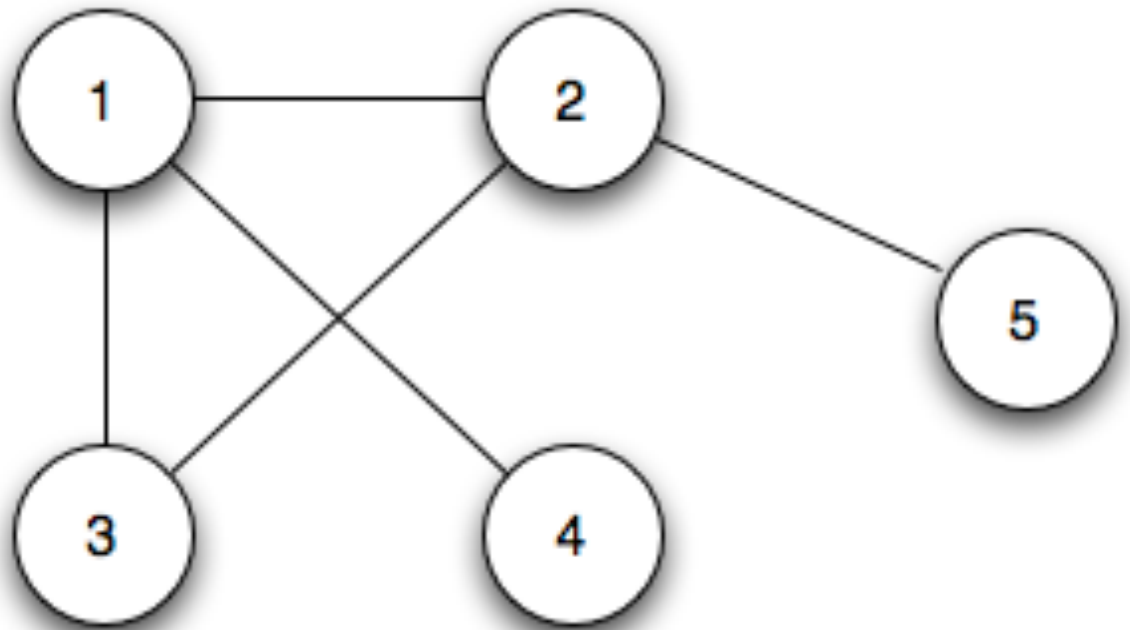
בהינתן $G = (E, V)$ ו-$s$ (קודקוד מקור), *BFS* בוחן את הקשתות כדי לגלות כל קודוקד שניתן להגיע אליו מ-$s$. הוא מחשב מרחק (מספר קשתות מינמלי) מ-$s$ לכל קודקוד ובונה "עץ רוחב" ששורשו $s$.
"רוחב" – האלגוריתם מגלה את כל הקודקודים שבמרחק $k$ מ-$s$ לפני שהוא מגלה את הקודקודים במרחק $k+1$ מ-$s$.

האלגוריתם:

- *Color* – מערך בגודל $|V|$ המאחסן את צבע הקודקוד
- $\Pi$ – מערך שמאחסן קודקוד קודם של קודקוד מסוים. לדוגמא - $\Pi[u]=v$ אומר שבמהלך הסריקה, $v$ נמצא לפני $u$.
- $d$ – מערך מרחקים מ-$s$.
- $Q$ – תור של קודקודים אפורים.

***BFS(E,V, s):***

```
for each u ∈ V-{s}
        color[u] = white
        d[u] = ∞
        Π[u] = null
color[s] = gray
d[s] = 0
Π[s] = null
Q.Enqueue(s)
while Q.IsEmpty ≠ false
        u = Q.Top
        for each v ∈ Adj[u]
                if color[v] == white
                        color[v] = gray
                        d[v] =  d[u]+1
                        Π[v] = u
                        Q.Enqueue(v)
        Q.Dequeue
        Color[u] = black
```

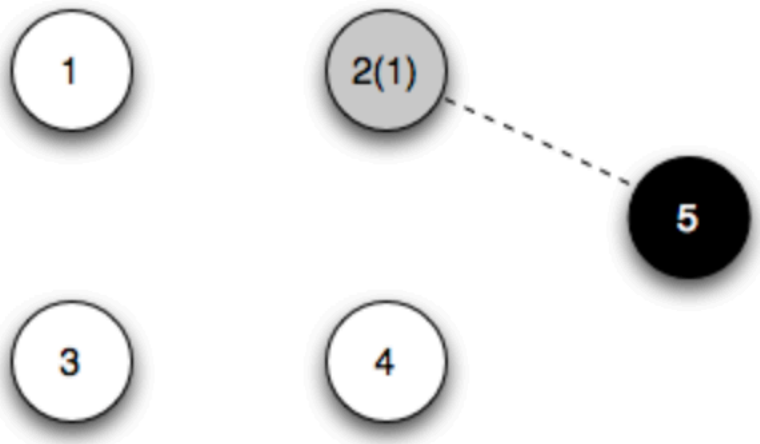

1
2
5
3
4

If we select vertex 5 as the source, then $d[5]=0$, $\pi[5]=/$, $Q=\{5\}$, thus the initialization gives

Q = {5}

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | ∞ | ∞ | ∞ | ∞ | 0 |
| π | / | / | / | / | / |

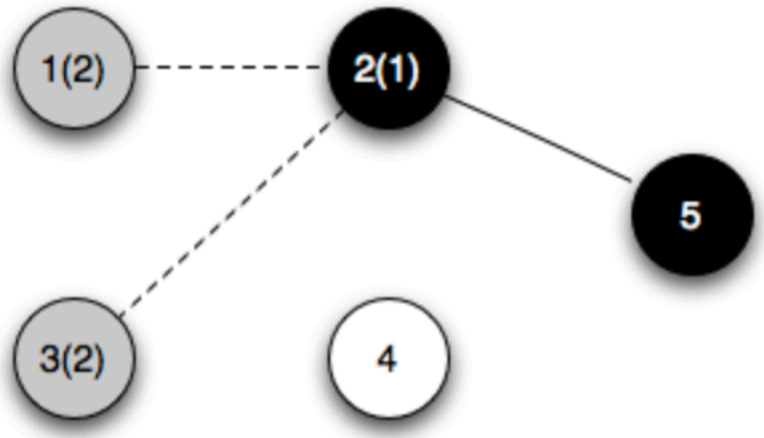
*Iteration 1*: dequeue vertex 5 and enqueue vertex 2




Q = {2}

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | ∞ | 1 | ∞ | ∞ | 0 |
| π | / | 5 | / | / | / |

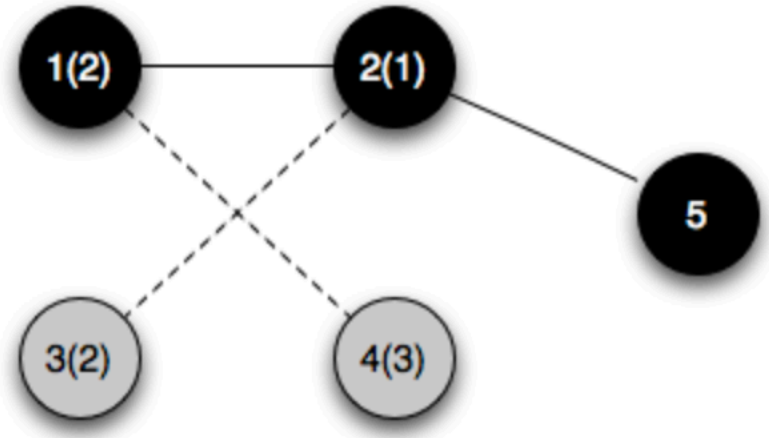
*Iteration 2*: dequeue vertex 2 and enqueue verticies 1,3




Q = {1,3}

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | 2 | 1 | 2 | ∞ | 0 |
| π | 2 | 5 | 2 | / | / |

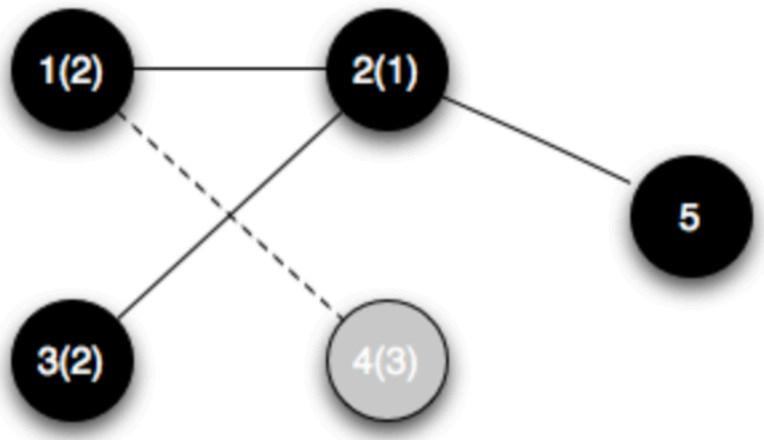
*Iteration 3*: dequeue vertex 1 and enqueue vertex 4




Q = {3,4}

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | 2 | 1 | 2 | 3 | 0 |
| π | 2 | 5 | 2 | 1 | / |

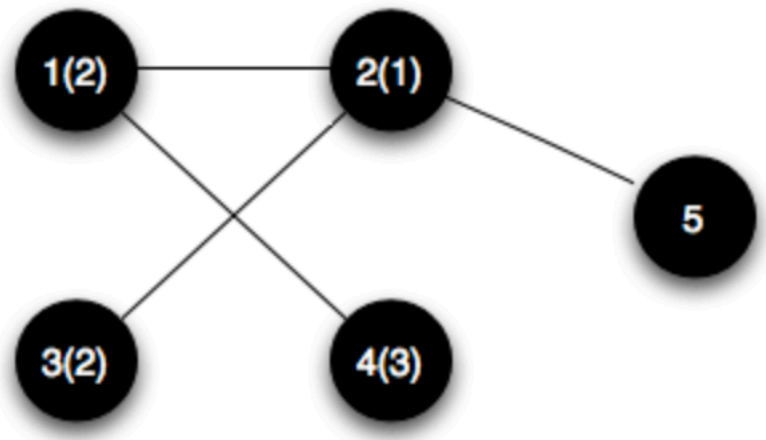
*Iteration 4*: dequeue vertex 3 and enqueue no vertices




Q = {4}

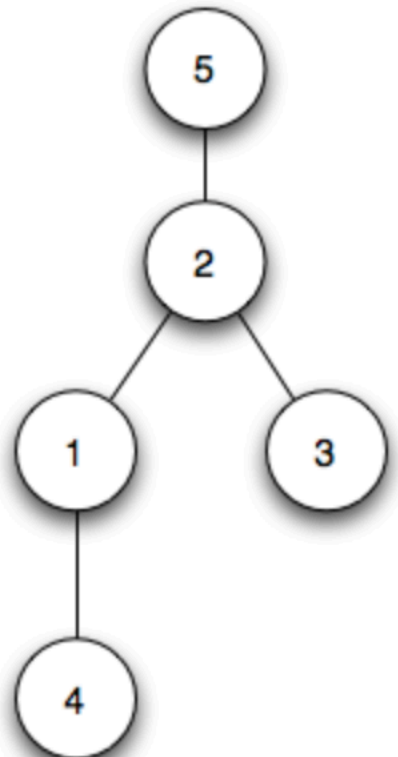
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | 2 | 1 | 2 | 3 | 0 |
| π | 2 | 5 | 2 | 1 | / |

*Iteration 5*: dequeue vertex 4 and enqueue no vertices (thus leaving the queue empty)




Q = ∅

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| d | 2 | 1 | 2 | 3 | 0 |
| π | 2 | 5 | 2 | 1 | / |


5
2
1
3
4

**חיפוש לעומק: *DFS***

מתחילים מקודקוד אקראי. נבדקות כל הקשתות היוצאות מן הקודקוד שהוא אחרון הקודקודים שהתגלו שעדיין יש לו קשתות שיוצאות ממנו ועדיין לא התגלו. לאחר שנבדקו כל הקשתות היוצאו מ-$v$, האלגוריתם "נסוג" וממשיך בבדיקת קשתות שיוצאות מהקודקוד שממנו התגלה $v$. (הרעיון דומה ל-*preorder* בעצים).

האלגוריתם:

- *Color* – מערך בגודל $|V|$ המאחסן את צבע הקודקוד. לבן- קודקוד שלא ביקרו בו, אפור – קודקוד שביקרו בו אך לא סיימנו לסרוק את הבנים שלו, שחור – סיימנו לסרוק את הבנים שלו.
- $\Pi$ – מערך שמאחסן קודקוד קודם של קודקוד מסוים. לדוגמא - $\Pi[u]=v$ אומר שבמהלך הסריקה, $v$ נמצא לפני $u$.
- $d$ – זמן המציין מתי הגענו לקודקוד מסוים בפעם הראשונה.
- $f$ – זמן סיום מעבר על כל הבנים של קודקוד מסוים.

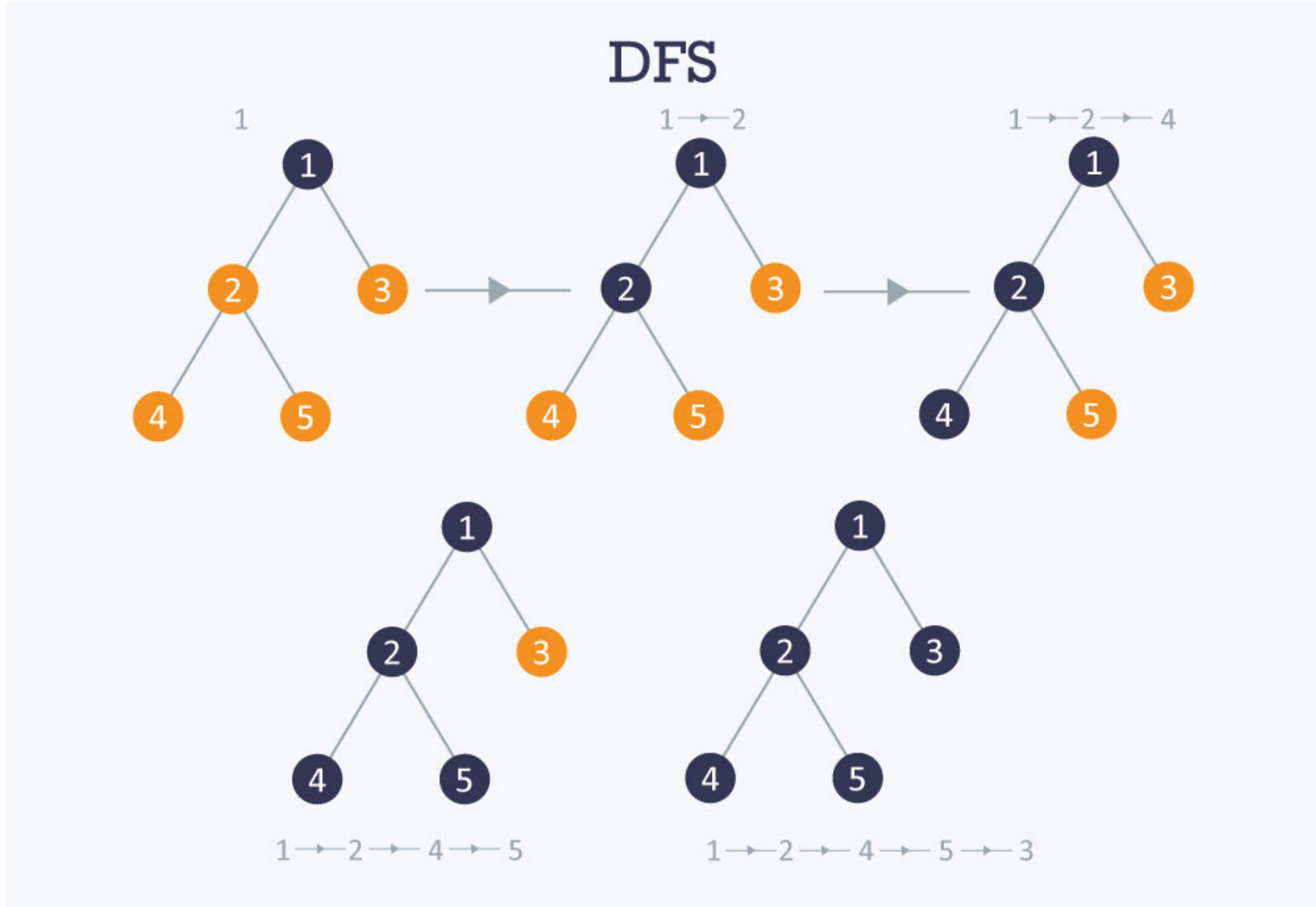
***DFS(V,E):***

```
for each u ∈ V
        color[u] = white
        Π[u] = null
time = 0
for each u ∈ V
        if color[u] == white
        DFS-visit(u)
```

***DFS-visit(u):***

```
color[u]=gray
time++
d[u]=time
for each v ∈ Adj[u]
        If color[v] == white
                Π[v]=u
                DFS-visit(v)
color[u] = black
f[u] = time++
```

סיבוכיות:- ***O(|V|+|E|)***


DFS
1
1 → 2
1 → 2 → 4
1 → 2 → 4 → 5
1 → 2 → 4 → 5 → 3

# תרגיל

- **הגדרה:** גרף לא מכוון $G=(V,E)$ יקרא גרף דו-צדדי (Bipartite graph) אם קיימת חלוקה של צמתי הגרף לקבוצות $L,R$ כך ש $G[L]$ ו $G[R]$ הם גרפים חסרי קשתות.
  - $L,R$ היא חלוקה של $V$. כלומר, $L \cup R = V$ וגם $L \cap R = \emptyset$.
  - הקבוצות L,R נקראות צדדים של הגרף.
- **תרגיל:** בהינתן גרף לא מכוון $G=(V,E)$ הציעו אלגוריתם המכריע
- אם $G$ הוא גרף דו צדדי.

# טענת עזר

- הוכיחו: גרף הינו דוצ אמם הוא לא מכיל מעגלים מאורך אי זוגי
- הוכחה:

נתחיל עם הכיוון הראשון (נניח גרף דו"צ ונראה שאינו מכיל מעגלים מאורך אי-זוגי). נניח בשלילה שהגרף כן מכיל מעגל באורך אי-זוגי, נניח $u_0 \to u_1 \to \ldots, u_{2k} \to u_{2k+1} = u_0$ (שימו לב שמספר הקשתות הוא אי-זוגי), ונניח בלי הגבלת כלליות ש-$u_0 \in V_0$. מכיוון שכל קשת שמתחילה ב-$V_0$ מסתיימת בקודקוד ב-$V_1$, ולהפך, נקבל כי - $u_1 \in V_1, u_2 \in V_0, \ldots, u_{2k} \in V_0$, ולכן $u_{2k+1} = u_0 \in V_1$, בסתירה לכך ש-$u_0 \in V_0$.

כעת, נניח שהגרף לא מכיל מעגל מאורך אי-זוגי. ההכוחה תהיה אלגוריתמית - בסעיף הבא נראה אלגוריתם המקבל גרף, ובמידה ואין בו מעגל מאורך אי-זוגי, האלגוריתם יחזיר שתי קבוצות $(V_0, V_1)$ שהם יהיו החלוקה של הקודקודים. הוכחה בסעיף הבא.

האלגוריתם יריץ ווריאציה על $BFS$. נתחיל בצומת שרירותי $s$. כל צומת במרחק זוגי מ-$s$ נכניס ל-$V_0$ וכל צומת במרחק אי-זוגי מ-$s$ נכניס ל-$V_1$. במידה וניסינו להכניס קודקוד שכבר מצאנו לקבוצה השנייה קיבלנו מעגל אי זוגי. נעצור ונחזיר "הגרך אינו דו צדדי".

כעת, בשביל להראות שהאלגוריתם נכון - נראה כי האלגוריתם יודיע "הגרף אינו דו צדדי" אם ורק אם קיים מעגל בגרף מעגל באורך אי-זוגי.

לכיוון הראשון, האלגוריתם מחזיר "הגרף אינו דו צדדי" אם קיים קודקוד $u$ שכבר הוכנס לאחת הקבוצות ורוצים להכניס אותו גם לקבוצה השנייה. מכיוון שהוכנס לקבוצה $V_0$ - קיים מסלול זוגי בין $s$ ל-$u$. בנוסף, מכיוון שהוכנס לקבוצה $V_1$, קיים מסלול אי זוגי בין $s$ לבין $u$. אם נשרשר את שני המסלולים, נקבל מסלול המתחיל ב-$s$ וחוזר אליו תוך מספר אי-זוגי של קשתות, ולכן קיים מעגל אי-זוגי.

בנוסף, קל לראות שברגע שיש מעגיל אי זוגי - נמצא אותו באלגוריתם.

<u>אתחול</u>:

נאתחל תור ריק Q, לכל $v \in V$ $d[v] = \infty$, ו- $U = W = \phi$.

כל עוד קיים צמת $s$ ב $V$ כך ש $d[s] = \infty$:

הכנס את $s$ לראש התור ואתחל $d[s] = 0$.

<u>כל עוד Q אינו ריק</u>:

הוצא את $v$ מראש התור. אם $d[v]$ זוגי הכנס את $v$ ל-$U$, אחרת הכנס אותו ל-$W$.

לכל צמת $u$ שכן של $v$ בצע:

- אם $d[u] = \infty$ אז הכנס את $u$ לסוף התור ו- $d[u] := d[v]+1$.
- אחרת אם $d[u]+ d[v]$ זוגי עצור ופלוט: "הגרף הוא לא דו צדדי!".

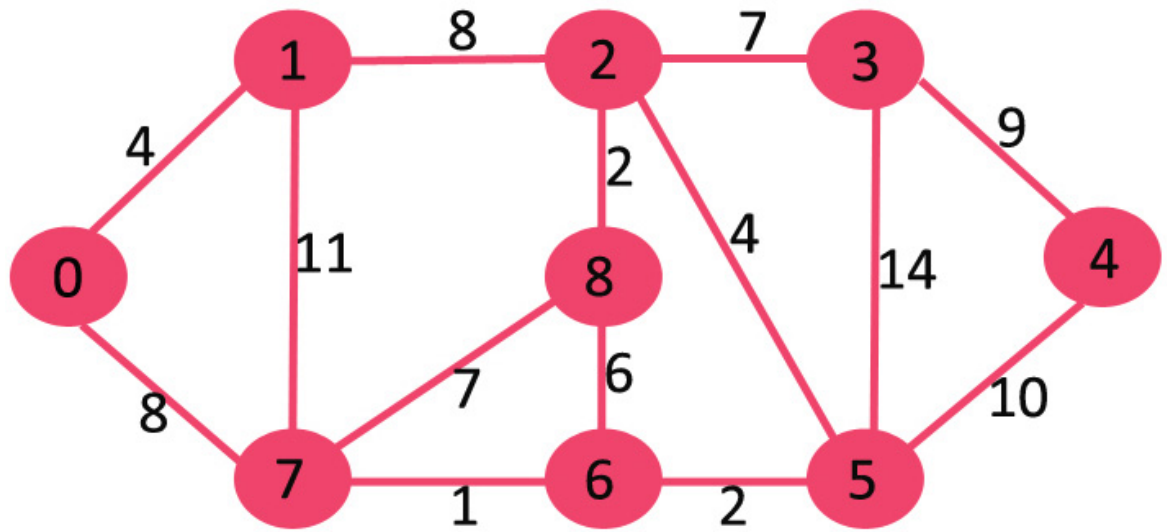
עץ – גרף קשיר ללא מעגלים, כלומר יש מסלול מכל קודקוד לכל קודקוד.

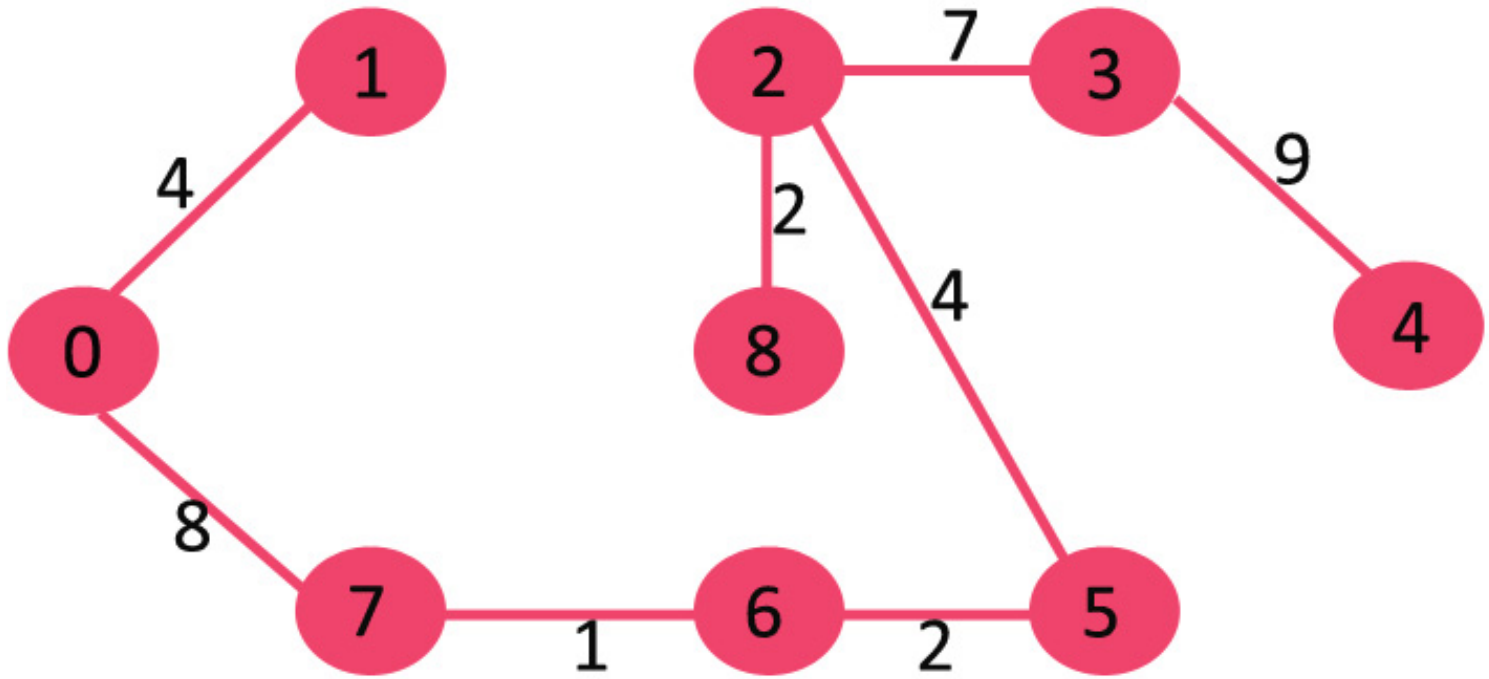
<u>הגדרה</u> – יהי גרף $G = (V, E)$ -פונקציית משקל על הקשתות $W: E \to \mathbb{R}$. <u>עץ פורש</u> הוא עץ (או תת-גרף) $T = (V, E')$ כאשר $E' \subseteq E$.
משקל $T$: $w(T) = \sum_{e \in E'} w(e)$.

עץ פורש מינימלי – עץ פורש עם המשקל הנמוך ביותר מבין כל העצים הפורשים של הגרף.

# אלגוריתם Kruskal

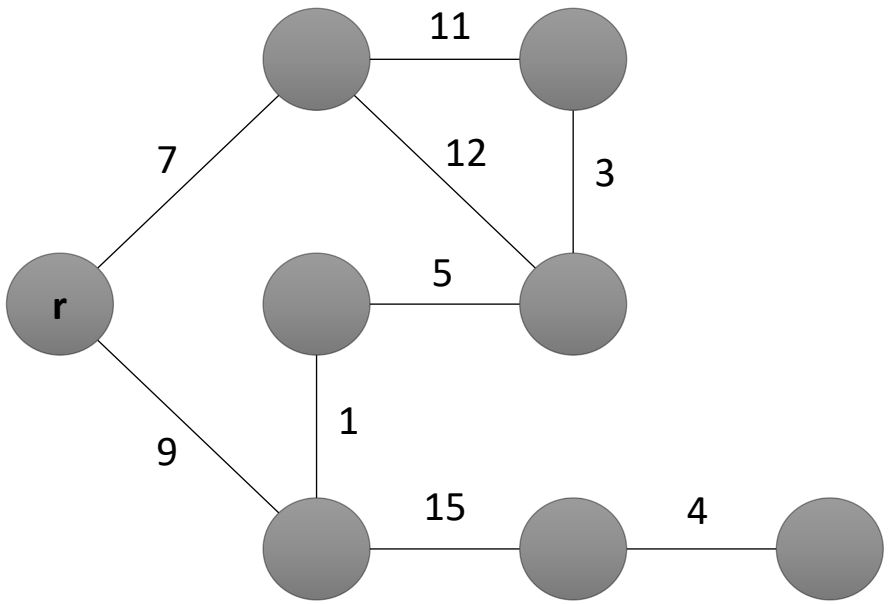
1. התחל מפתרון ריק $F \leftarrow \emptyset$.
2. מיין את קשתות הגרף לפי משקלן $w(e)$.
3. לכל קשת $e \in E$ בסדר זה
   3.1. אם הקשת לא סוגרת מעגל בגרף $(V,F)$, הוסף את $e$ ל $F$
   3.2. אחרת, אל תבצע

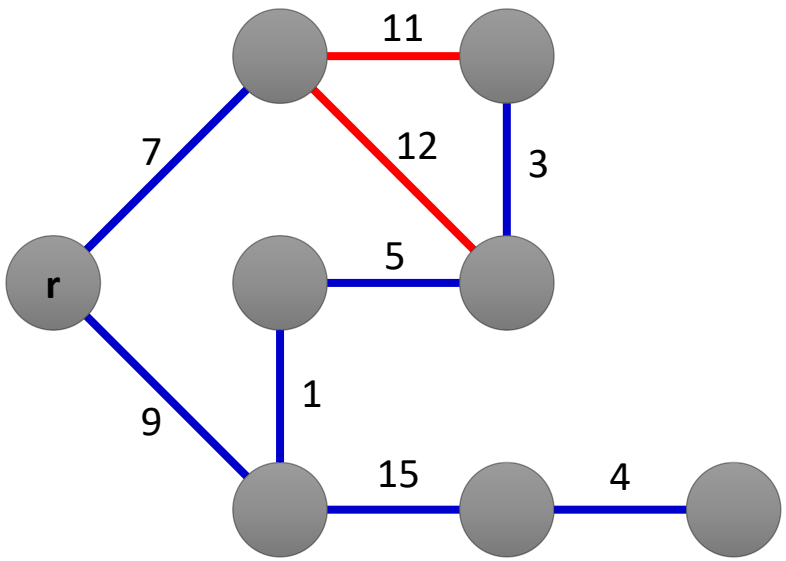

0
1
2
3
4
5
6
7
8
4
8
11
8
7
2
4
7
14
9
6
10
1
2


1
2
7
3
4
9
2
0
8
4
4
8
7
1
6
2
5

# אלגוריתם Prim

1. התחל מצומת $r$ כלשהו וקבע $T \leftarrow \{r\}$.
2. כל עוד $T \neq V$
   1. מצא את הקשת $(u,v)$ הקלה ביותר היוצאת מ $T$ ($u \in T, v \notin T$).
   2. בחר את הקשת לעץ
   3. (זרוק את כל הקשתות $(w,v)$ עבור $w \in T$ )


11
7
12
3
r
5
1
9
15
4


11
7
12
3
r
5
1
9
15
4

**תרגיל:**

נתון גרף לא מכוון $G = (V, E)$ פונקציית משקל על הקשתות $w: E \to \mathbb{R}$. יהי $T$ עפ"מ של $G$ לפי פונקצית המשקל $w$. נגדיר פונקציית משקל חדשה $w': E \to \mathbb{R}$ באופן הבא: $w'(e) = w(e) + c$ כאשר $c \in \mathbb{R}$ קבוע כלשהו. האם $T$ הוא עפ"מ גם לפי $w'$?

פיתרון:

נוכיח שכן – נניח בשלילה כי קיים עפ"מ $T'$ כך ש- $\sum_{e \in T'} w'(e) < \sum_{e \in T} w'(e)$. מספר הקשתות ב-$T$ וב-$T'$ הוא בדיוק $|V|-1$. לכן:

$$\sum_{e \in T'} w(e) + c(|V| - 1) = \sum_{e \in T'} w'(e) < \sum_{e \in T} w'(e) = \sum_{e \in T} w(e) + c(|V| - 1)$$

לכן:

$$\sum_{e \in T'} w(e) < \sum_{e \in T} w(e)$$

וזו סתירה לכך ש-T הוא עפ"מ לפי w.

**תרגיל:**

נתון גרף $G = (V, E)$ לא מכוון וקשיר. כמו כן נתונות פונקציות משקל על הקשתות $w: E \to \mathbb{R}$, $w': E \to \mathbb{R}$ המקיימות $w(e_1) \leq w(e_2)$ אם"ם $w'(e_1) \leq w'(e_2)$ לכל זוג קשתות $e_1, e_2$. הוכח כי $T$ הוא עפ"מ לפי $w'$ אם"ם הוא עפ"מ לפי $w$.

הוכחה:

← יהי $T$ עפ"מ לפי $w$. לכן קיימת ריצה של קרוסקל המוצאת את $T$ לפי $w$ (ללא הוכחה). כלומר קיים סידור מונוטוני לא יורד של הקשתות לפי $w$ שהרצת קרוסקל עליו מניבה את $T$. מהנחת התרגיל, סידור זה הוא מונוטוני לא יורד גם לפי $w'$, ולכן מנכונות קרוסקל $T$ הוא עפ"מ לפי $w'$.

→ ההוכחה זהה כאשר מחליפים את $w, w'$.

<u>מסקנה</u> – בבניית עפ"מ, מה שמשנה הוא סדר המשקלות ולא המשקלות עצמן.