מגיש:  
מיכאל לוגסי  
305536575

# אלגוריתמים מבוזרים תרגיל 1

שאלה 1:

**סעיף א:**  
***צומת parent*** *הוא הצומת הראשון שממנו התקבלה הודעת קישור (למשל "CONNECT") לצומת הנוכחי. הצומת הנוכחי מאמץ אותו כצומת הורה בעץ הפורש ומשיב לו "ACK".*

***צומת child*** *הוא כל צומת שכנה אשר שלחה הודעת "ACK" בחזרה לצומת הנוכחי, בעקבות כך שהוא שלח אליה בקשת "CONNECT". כלומר, כל שכן שקיבל את הצומת הנוכחי כהורה בעץ, נחשב ל־child שלו.*

*כלומר, קשרי הורה-ילד נוצרים באופן חד-כיווני על בסיס קבלת הודעת "CONNECT" ראשונה, ו־ACK חזרה, תוך שמירה על כך שלא ייווצרו מחזורים.*

**סעיף ג':**  
סיבוכיות הודעות היא - O(m) - כאשר m הוא מספר הקשתות בגרף (בכל קשת עוברת הודעה אחת של "CONNECT" ואחת נוספת של "ACK" או "NACK"  
*זמן הריצה הוא O(D), כאשר D הוא הקוטר של הגרף  
כלומר, המספר המקסימלי של קפיצות בין שורש (root) לצומת כלשהו בגרף*

**סעיף ב':**

*class Node:*

*def \_\_init\_\_(self, name: str, neighbors: list[str]):*

*self.Name = name*

*self.Neighbors = neighbors*

*self.Parent = None*

*self.Childs = []*

*self.Got\_rsp = {neighbor: False for neighbor in neighbors}*

*def send\_msg(self, msg: str, dest: str):*

*# sending a message to dest*

*def build(self):*

*self.Parent = None*

*for node in self.Neighbors:*

*self.send\_msg("CONNECT", node)*

*def on\_receive\_msg(self, msg: str, src: str):*

*if msg == "CONNECT":*

*self.on\_receive\_connect(src)*

*elif msg == "ACK":*

*self.on\_receive\_ack(src)*

*elif msg == "NACK":*

*self.on\_receive\_nack(src)*

*def on\_receive\_connect(self, src: str):*

*self.Got\_rsp[src] = True*

*if self.Parent is None:*

*self.Parent = src*

*for node in self.Neighbors:*

*if node != src:*

*self.send\_msg("CONNECT", node)*

*else:*

*self.send\_msg("NACK", src)*

*def on\_receive\_ack(self, src: str):*

*self.Got\_rsp[src] = True*

*self.Childs.append(src)*

*if self.got\_rsp\_from\_all() and self.Parent is not None:*

*self.send\_msg("ACK", self.Parent)*

*def on\_receive\_nack(self, src: str):*

*self.Got\_rsp[src] = True*

*def got\_rsp\_from\_all(self) -> bool:*

*for src in self.Got\_rsp:*

*if not self.Got\_rsp[src]:*

*return False*

*return True*

*def terminate(self) -> bool:*

*if self.got\_rso\_from\_all:*

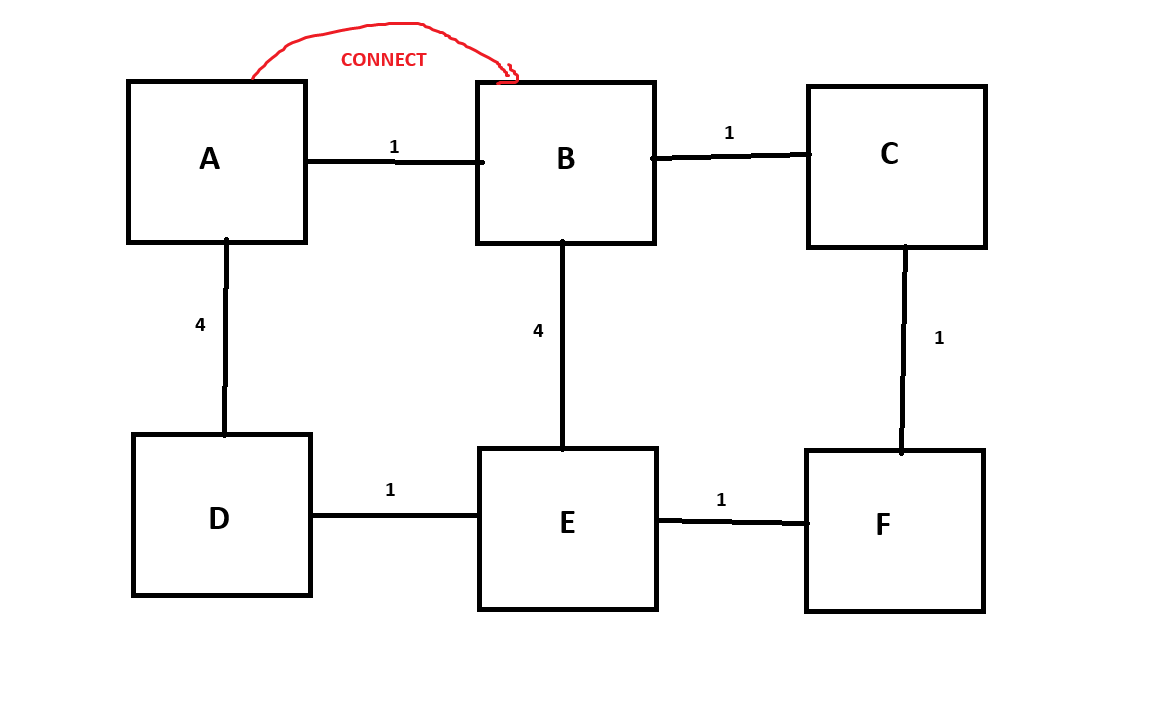
*#DO TERMINATE*

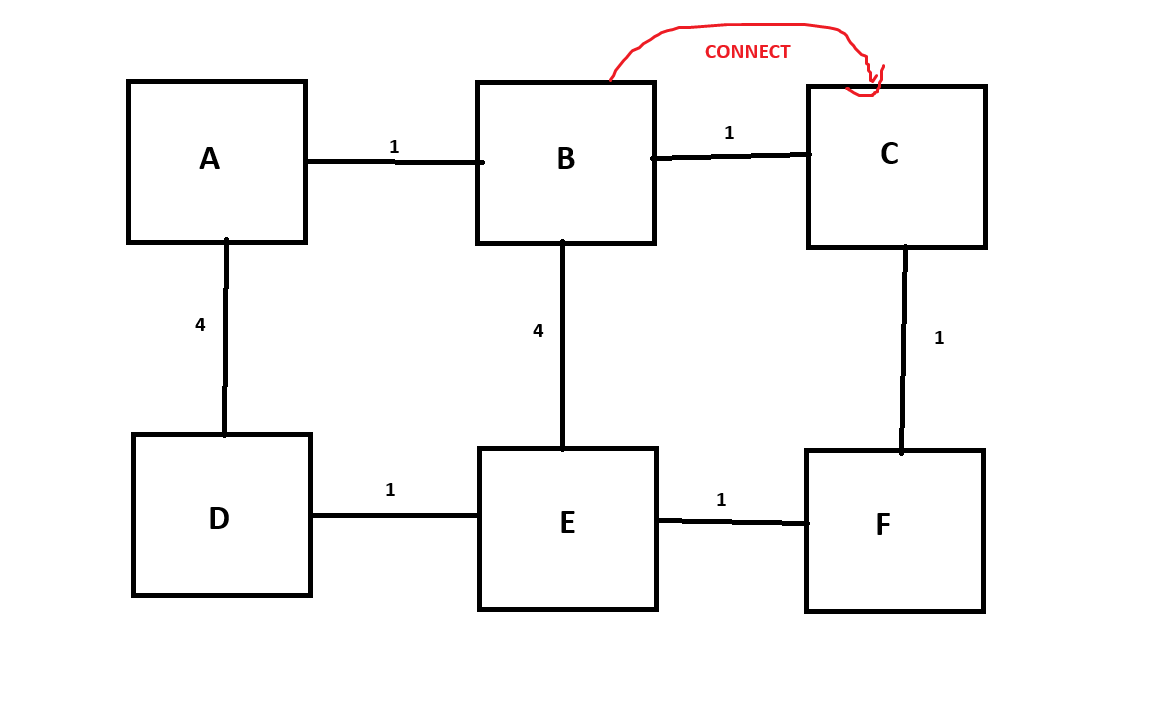
*return True*

*return False*

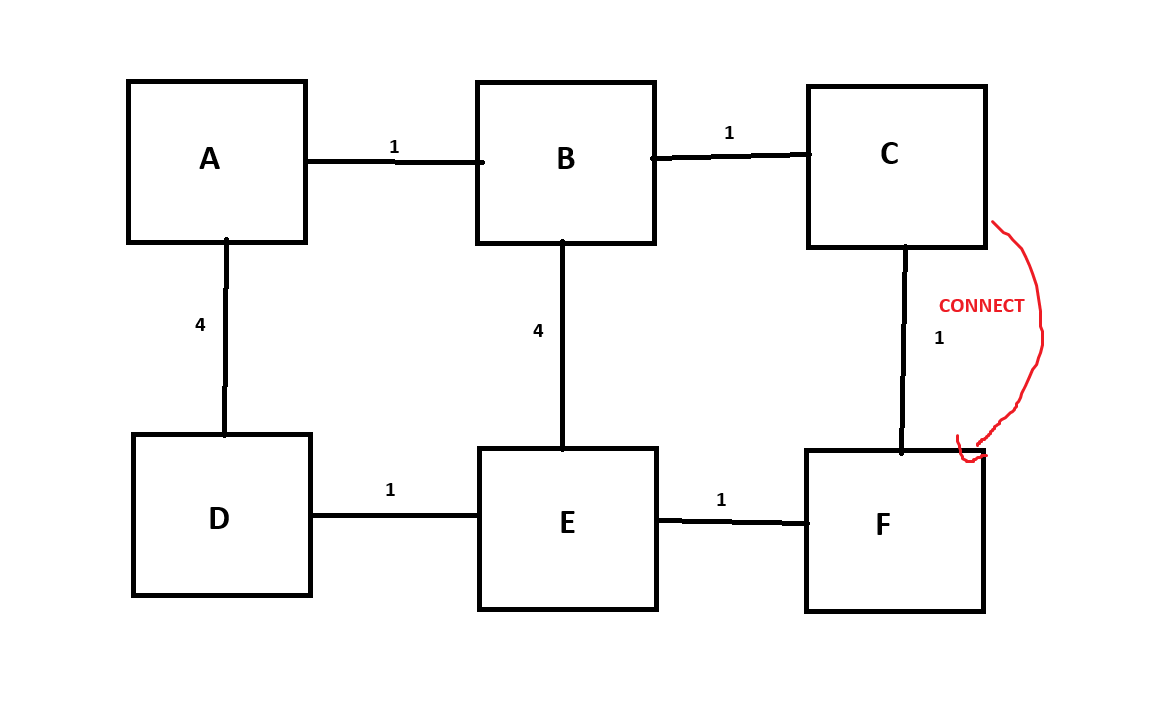
**סעיף ד':**

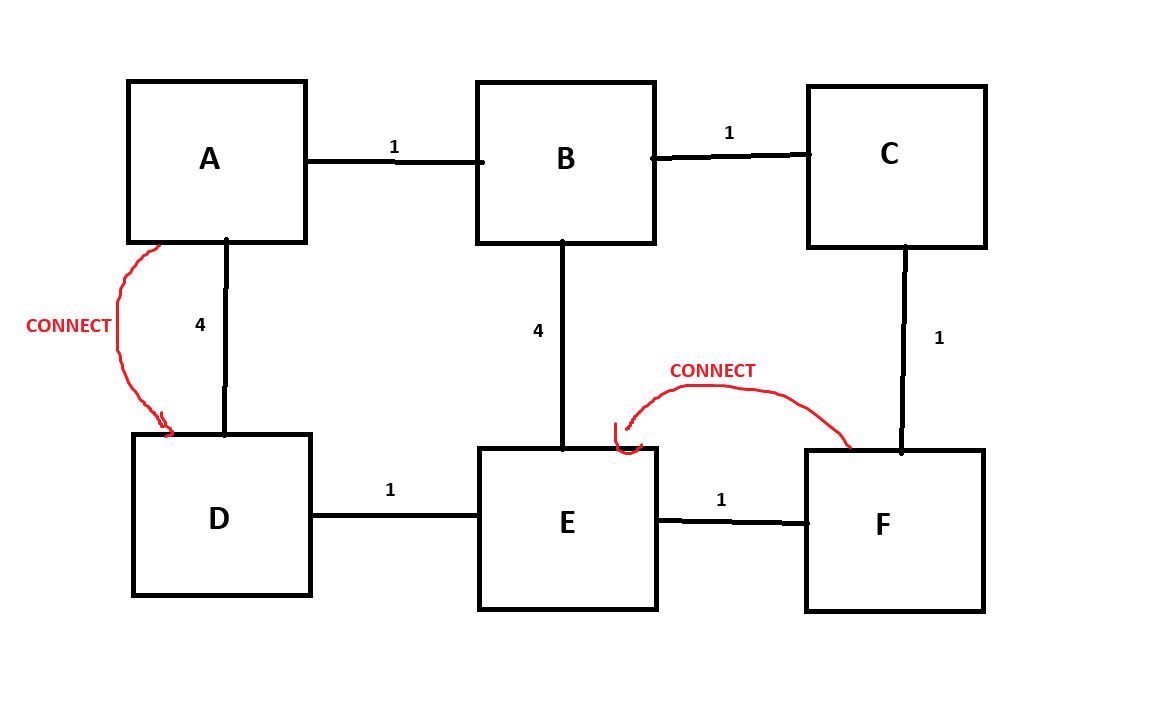
זמן 1:



זמן 2:  


זמן 3:

  
זמן 4:

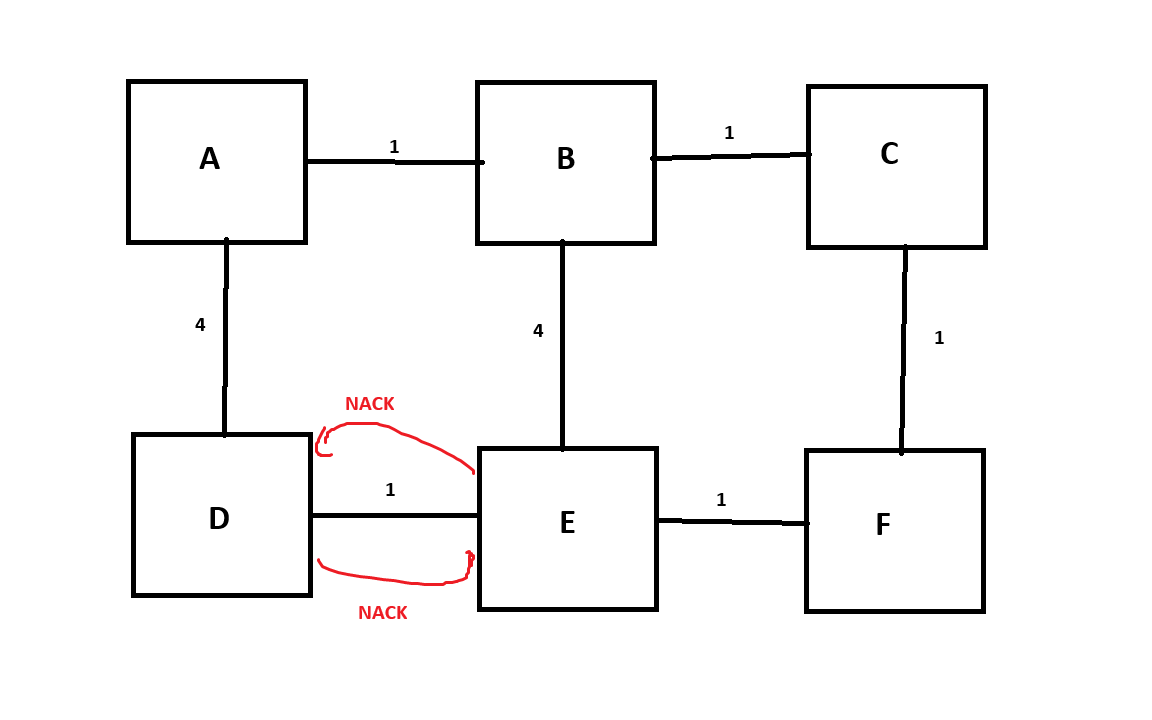


זמן 5:

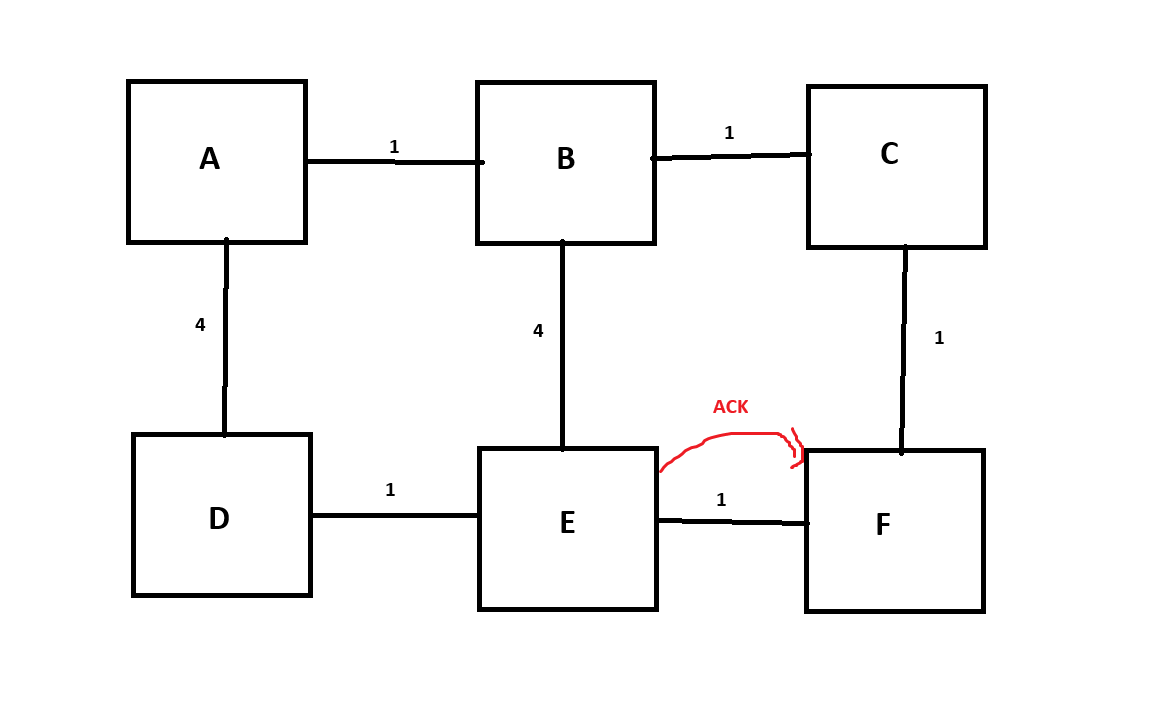
A diagram of a connection

AI-generated content may be incorrect.

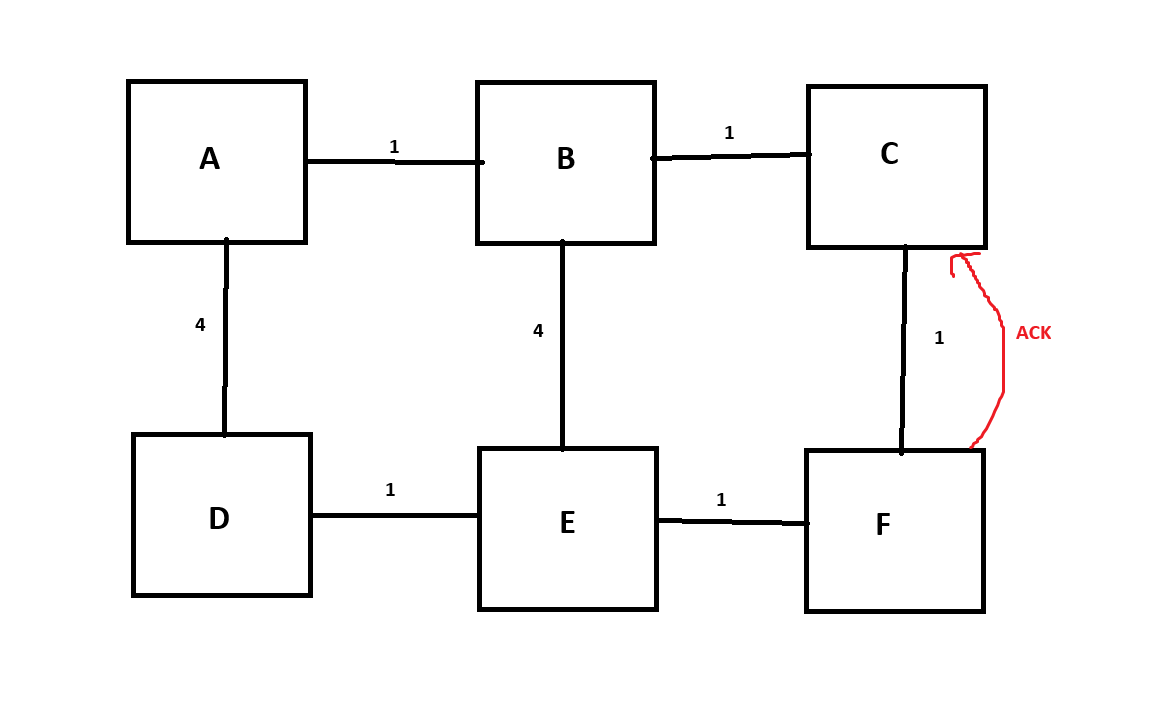
זמן 6:



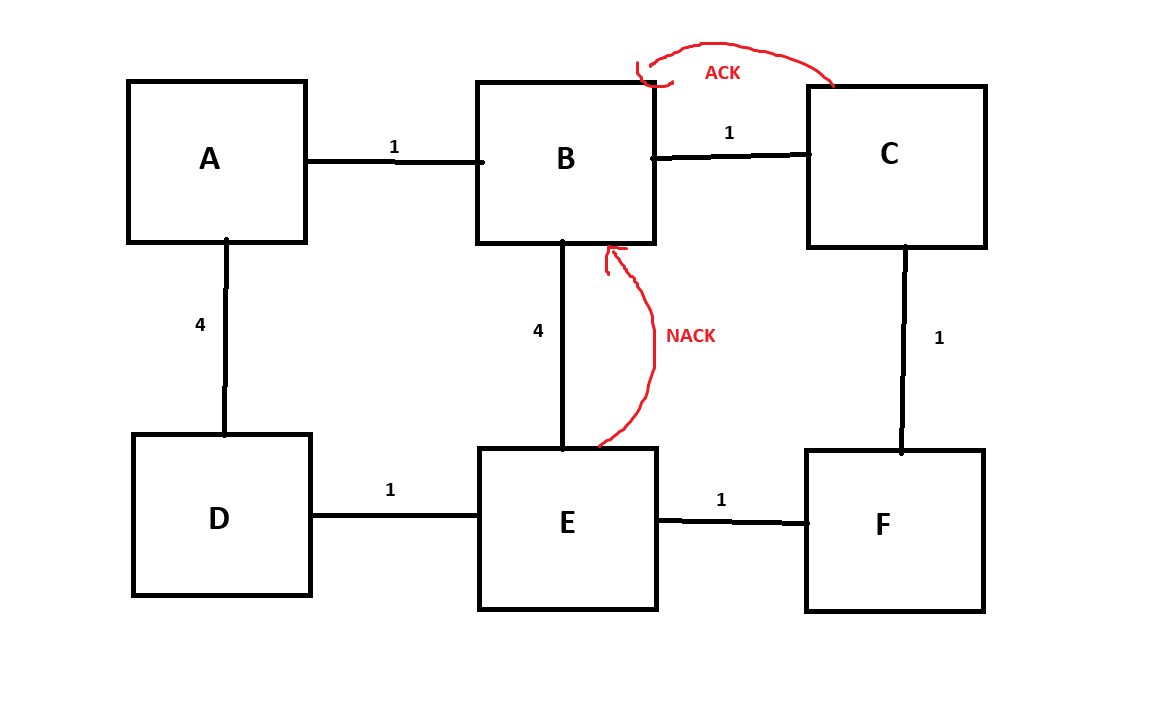
זמן 7:



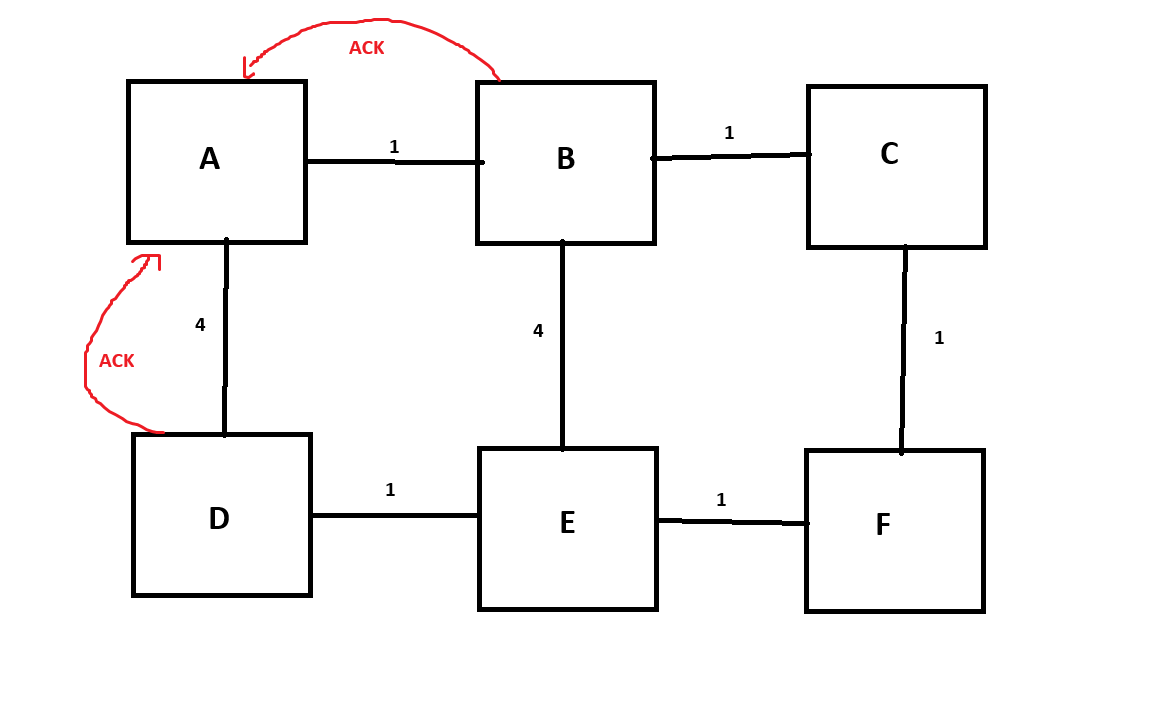
זמן 8:



זמן 9:



זמן 10:



שאלה 2:

סעיף א:

class Router:

def \_\_init\_\_(self, name: str, all\_nodes: list[str], neighbors: dict):

self.name = name

self.all\_nodes = all\_nodes

self.neighbors = neighbors # neighbors: dict[str, int] - maps neighbor name to cost

self.routing\_table = {}

self.outbuf = {neighbor: [] for neighbor in neighbors}

self.inbuf = {neighbor: [] for neighbor in neighbors}

self.updated = False

self.build\_routing\_table()

self.broadcast\_routing\_table()

def build\_routing\_table(self):

all\_nodes = self.all\_nodes

for dest in all\_nodes:

self.routing\_table[dest] = {}

for via in all\_nodes:

if dest == via:

if dest == self.name:

self.routing\_table[dest][via] = 0

elif via in self.neighbors:

self.routing\_table[dest][via] = self.neighbors[via]

else:

self.routing\_table[dest][via] = float('inf')

elif via == self.name:

self.routing\_table[dest][via] = None # can't route via self

else:

self.routing\_table[dest][via] = float('inf')

def receive\_update(self, msg: dict, src: str):

for dest, cost in msg.items():

new\_cost = cost + self.neighbors[src] # Use actual link cost

previous\_cost = self.routing\_table[dest][src]

if new\_cost != previous\_cost:

self.routing\_table[dest][src] = new\_cost

self.updated = True

if self.updated:

self.broadcast\_routing\_table()

self.updated = False

def best\_routes(self) -> dict:

best = {}

for dest, via\_map in self.routing\_table.items():

valid\_routes = {via: cost for via, cost in via\_map.items() if cost is not None and cost < float('inf')}

if valid\_routes:

best\_via = min(valid\_routes.items(), key=lambda item: item[1])

best[dest] = best\_via[1]

return best

def broadcast\_routing\_table(self):

best\_routes = self.best\_routes()

for neighbor in self.neighbors:

self.outbuf[neighbor].append(best\_routes)

def send\_msg(self, dest: str, msg: dict):

# Placeholder for actual message-passing mechanism

pass

def receive\_from(self, src: str, msg: dict):

self.inbuf[src].append(msg)

self.receive\_update(msg, src)

**סעיף ב':**

ראוטר A יכול להתעדכן רק משכניו B ו D   
במקרה שלנו A יעדיף לעבור דרך שכן B בשביל להגיע לקודקוד F – להלן שלבי התעדכנות של A  
בטלבאות בכל מקום שיש תא רק הכוונה שיש שם ערך של אינסוף.

חישובים ש A עושה לפי הנוסחא הבא:  
D(A,X) = min { C(A,B) + D(B,X), C(A,D) + D(D,X) }

D(A,B) = min { C(A,B), D(A,D) + C(D,B) }

D(A,D) = min { C(A,D), D(A,B) + C(B,D) }

**זמן 1:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **From A** | Via A | Via B | Via C | Via  D | Via E | Via F |
| To A | NA | NA | NA | NA | NA | NA |
| To B | NA | 1 |  |  |  |  |
| To C | NA |  |  |  |  |  |
| To D | NA |  |  | 4 |  |  |
| To E | NA |  |  |  |  |  |
| To F | NA |  |  |  |  |  |

בטבלה הערכים של קודקוד A לאחר חישובים  
העידכונים שכל צומת תשלח לאחר חישוב של כל אחד

* **צומת A-**   
  **D(A,B) = 1**, **D(A,D) = 4** – כל השאר אינסוף
* **צומת B -  
  D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 4** – כל השאר אינסוף
* **צומת C -  
  D(C,B) = 1, D(C,F) = 1** – כל השאר אינסוף
* **צומת D** -  
  **D(D,A) = 4, D(D,E) = 1** – כל השאר אינסוף
* **צומת E** -  
  **D(E,D) = 1, D(E,B) = 4, D(E,F) = 1** – כל השאר אינסוף
* **צומת F** -  
  **D(F,C) = 1, D(F,E) = 1** – כל השאר אינסוף

**זמן 2:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **From A** | Via A | Via B | Via C | Via  D | Via E | Via F |
| To A | NA | NA | NA | NA | NA | NA |
| To B | NA | 1 |  |  |  |  |
| To C | NA | 2 |  |  |  |  |
| To D | NA |  |  | 4 |  |  |
| To E | NA | 5 |  | 5 |  |  |
| To F | NA |  |  |  |  |  |

בטבלה הערכים של קודקוד A לאחר חישובים  
לאחר העדכונים להלן החישובים שצומת A תבצע לאחרת העידכונים הבאים   
מצומת B - **D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 4** מצומת D - **D(D,A) = 4, D(D,E) = 1**

* + ‎ **D(A,B) = min { 1, 4 + INF } = 1**
  + **D(A,C) = min { 1 + 1, 4 + INF } = 2**
  + **D(A,D) = min { 4, 1 + INF } = 4**
  + **D(A,E) = min { 1 + 4, 4 + 1 } = 5**

העידכונים שכל צומת תשלח לאחר חישוב של כל אחד:

* **צומת A-**   
  **D(A,B) = 1**, **D(A,C)=2**, **D(A,D) = 4**, **D(A,E) = 5** - כל השאר אינסוף
* **צומת B -  
  D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 4**, **D(B,F) = 2**, **D(B,D) = 5**
* **צומת C -  
  D(C,B) = 1**, **D(C,F) = 1**, **D(C,A) = 2**, **D(C,E) = 2** - כל השאר אינסוף
* **צומת D** -  
  **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 5**, **D(D,F) = 2** - כל השאר אינסוף
* **צומת E** -  
  **D(E,D) = 1**, **D(E,B) = 4**, **D(E,F) = 1**, **D(E,A) = 5**, **D(E,C) = 2**
* **צומת F** -  
  **D(F,C) = 1**, **D(F,E) = 1**, **D(F,B) = 2**, **D(F,D) = 2** - כל השאר אינסוף

**זמן 3:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **From A** | Via A | Via B | Via C | Via  D | Via E | Via F |
| To A | NA | NA | NA | NA | NA | NA |
| To B | NA | 1 |  | 9 |  |  |
| To C | NA | 2 |  |  |  |  |
| To D | NA | 6 |  | 4 |  |  |
| To E | NA | 5 |  | 5 |  |  |
| To F | NA | 3 |  | 6 |  |  |

בטבלה הערכים של קודקוד A לאחר חישוב

לאחר העדכונים להלן החישובים שצומת A תבצע לאחרת העידכונים הבאים   
מצומת B - **D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 4**, **D(B,F) = 2**, **D(B,D) = 5**מצומת D - **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 5**, **D(D,F) = 2**

* + ‎ **D(A,B) = min { 1, 4 + 5 } = 1**
  + **D(A,C) = min { 1 + 1, 4 + INF } = 2**
  + **D(A,D) = min { 4, 1 + 5 } = 4**
  + **D(A,E) = min { 1 + 4, 4 + 1 } = 5**
  + **D(A,F) = min { 1 + 2, 4 + 2 } = 3**

העידכונים שכל צומת תשלח לאחר חישוב של כל אחד:

* **צומת A-**   
  **D(A,B) = 1**, **D(A,C)=2**, **D(A,D) = 4**, **D(A,E) = 5**, **D(A,F) = 3**
* **צומת B -  
  D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 3**, **D(B,F) = 2**, **D(B,D) = 5**
* **צומת C -  
  D(C,B) = 1**, **D(C,F) = 1**, **D(C,A) = 2**, **D(C,D) = 3**, **D(C,E) = 2**
* **צומת D** -  
  **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 5, D(D,C) = 3**, **D(D,F) = 2**
* **צומת E** -  
  **D(E,D) = 1**, **D(E,B) = 3**, **D(E,F) = 1**, **D(E,A) = 5**, **D(E,C) = 2**
* **צומת F** -  
  **D(F,C) = 1**, **D(F,E) = 1**, **D(F,B) = 2**, **D(F,D) = 2, D(F,A) = 3**

**זמן 4:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **From A** | Via A | Via B | Via C | Via  D | Via E | Via F |
| To A | NA | NA | NA | NA | NA | NA |
| To B | NA | 1 |  | 9 |  |  |
| To C | NA | 2 |  | 7 |  |  |
| To D | NA | 6 |  | 4 |  |  |
| To E | NA | 4 |  | 5 |  |  |
| To F | NA | 3 |  | 6 |  |  |

בטבלה הערכים של קודקוד A לאחר חישוב  
לאחר העדכונים להלן החישובים שצומת A תבצע לאחרת העידכונים הבאים   
מצומת B - **D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 3**, **D(B,F) = 2**, **D(B,D) = 5**מצומת D - **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 5, D(D,C) = 3**, **D(D,F) = 2**

* + ‎ **D(A,B) = min { 1, 4 + 5 } = 1**
  + **D(A,C) = min { 1 + 1, 4 + 3 } = 2**
  + **D(A,D) = min { 4, 1 + 5 } = 4**
  + **D(A,E) = min { 1 + 3, 4 + 1 } = 4**
  + **D(A,F) = min { 1 + 2, 4 + 2 } = 3**

העידכונים שכל צומת תשלח לאחר חישוב של כל אחד:

* **צומת A-**   
  **D(A,B) = 1**, **D(A,C)=2**, **D(A,D) = 4**, **D(A,E) = 5**, **D(A,F) = 3**
* **צומת B -  
  D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 3**, **D(B,F) = 2**, **D(B,D) = 4**
* **צומת C -  
  D(C,B) = 1**, **D(C,F) = 1**, **D(C,A) = 2**, **D(C,D) = 3**, **D(C,E) = 2**
* **צומת D** -  
  **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 4, D(D,C) = 3**, **D(D,F) = 2**
* **צומת E** -  
  **D(E,D) = 1**, **D(E,B) = 3**, **D(E,F) = 1**, **D(E,A) = 4**, **D(E,C) = 2**
* **צומת F** -  
  **D(F,C) = 1**, **D(F,E) = 1**, **D(F,B) = 2**, **D(F,D) = 2, D(F,A) = 3**

**זמן 5:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **From A** | Via A | Via B | Via C | Via  D | Via E | Via F |
| To A | NA | NA | NA | NA | NA | NA |
| To B | NA | 1 |  | 8 |  |  |
| To C | NA | 2 |  | 7 |  |  |
| To D | NA | 5 |  | 4 |  |  |
| To E | NA | 4 |  | 5 |  |  |
| To F | NA | 3 |  | 6 |  |  |

בטבלה הערכים של קודקוד A לאחר חישוב  
לאחר העדכונים להלן החישובים שצומת A תבצע לאחרת העידכונים הבאים   
מצומת B - **D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 3**, **D(B,F) = 2**, **D(B,D) = 4** מצומת D - **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 4, D(D,C) = 3**, **D(D,F) = 2**

* + ‎ **D(A,B) = min { 1, 4 + 4 } = 1**
  + **D(A,C) = min { 1 + 1, 4 + 3 } = 2**
  + **D(A,D) = min { 4, 1 + 4 } = 4**
  + **D(A,E) = min { 1 + 3, 4 + 1 } = 4**
  + **D(A,F) = min { 1 + 2, 4 + 2 } = 3**

העידכונים שכל צומת תשלח לאחר חישוב של כל אחד:

* **צומת A-**   
  **D(A,B) = 1**, **D(A,C)=2**, **D(A,D) = 4**, **D(A,E) = 5**, **D(A,F) = 3**
* **צומת B -  
  D(B,A) = 1**, **D(B,C) = 1**, **D(B,E) = 3**, **D(B,F) = 2**, **D(B,D) = 4**
* **צומת C -  
  D(C,B) = 1**, **D(C,F) = 1**, **D(C,A) = 2**, **D(C,D) = 3**, **D(C,E) = 2**
* **צומת D** -  
  **D(D,A) = 4**, **D(D,E) = 1**, **D(D,B) = 4, D(D,C) = 3**, **D(D,F) = 2**
* **צומת E** -  
  **D(E,D) = 1**, **D(E,B) = 3**, **D(E,F) = 1**, **D(E,A) = 4**, **D(E,C) = 2**
* **צומת F** -  
  **D(F,C) = 1**, **D(F,E) = 1**, **D(F,B) = 2**, **D(F,D) = 2, D(F,A) = 3**

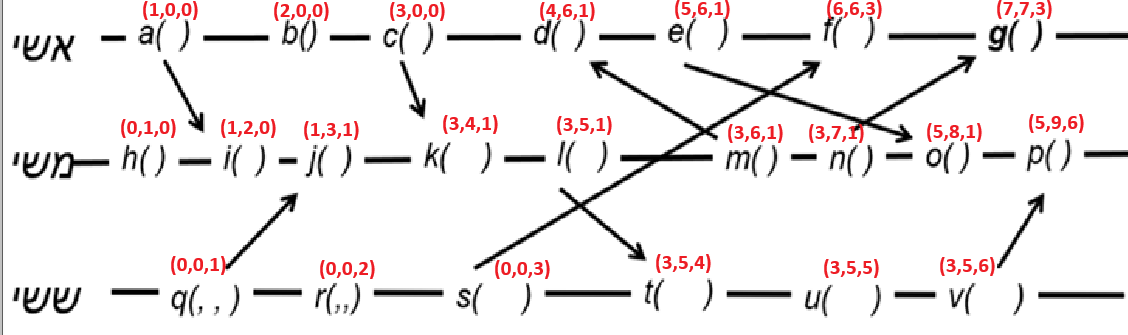
לאחר זמן זה יתבעו שוב ושוב עידכונים אבל שום דבר לא ישתנה אלא אם כן יקרה משהו במערכת.  
לכן ניתן לומר ש A יעביר מידע ל F דרך קודקוד B מאחר והעלות של מעבר ל F דרך B נמוך יותר.

**סעיף ג':**

ברגע השינוי וברגע שקודקודים B ו C יתוודעו לשינוי הן יבצעו עידכון למשקולות שלהן שיחלחלו ברשת כמו שראינו בסעיף הקודם, רק שעלול להיווצר לופים מאוד ארוכים עד שהערכים יתעדכו עם הערכים הנכונים מאחר שלדוגמא, כשקודקוד C ירצה לעדכן את העלות מסלול שלו ל B עם העלות החדשה שנוצרה הוא יבצע את הנוסחא הבא:  
D(C,B) = min { D(C,B) , D(C,F) + D(F,B) } = {15, 1 + 2} = 3  
  
שזו כמובן טעות, בעצם C לא יודע ש F עובר ל B דרך C עצמו ובעצם גם הוא מושפע מהשינוי הזה.  
ואז יווצר לופ כי C יעדכן את F שהעלות החדשה היא 3 ואז F יעדכן את העלות החדשה שלו ל B 4 וכך זה יחזור עד שיגיע לערכים הנכונים.  
הפתרון הוא שכל צומת שמפרסת את העלויות שלה תעביר ערך אינסוף לשכנים שלה, במקומות שבהם הדרך שלהם אל היעד עוברת דרך השכן ששלחתי לו.  
במקרה שלנו F לא יגיד ל C שהעלות שלו ל B היא 2 אלא יפרסם ל C שהעלות ל B היא אינסוף מאחר ו F עובר דרך C כדי להגיע ל B, אך ל E הוא יפרסם שהעלות היא ב 2 מאחר ו F מגיע ל B לא דרך E.  
וזה יפתור את הבעיה מאחר ובמקרה שלנו הנוסחא תהיה כעת:  
D(C,B) = min { D(C,B) , D(C,F) + D(F,B) } = {15, 1 + INF} = 15

שזה בדיוק מה שהיה צריך להתבצע ובכך חסכנו לופים רבים.

שאלה 3:  
**סעיף א:**

  
  
**סעיף ב':**  
ניתן לראות של אשי חסרים 2 אירועים של משי ו 3 אירועים של ששי.  
למשי חסרים 2 אירועים של אשי ולא חסר שום אירוע של ששי.  
לששי חסרים 4 אירועים של אשי ו 4 אירועים של משי.  
לכן ניתן להגיד, שאין אפליקציה שמכירה את כל האירועים, אך ניתן להגיד שמשי מכירה הכי הרבה אירועים לאחר מכן אשי ולאחר מכן ששי.

**סעיף ג':**

טבלה של אשי

|  |  |  |
| --- | --- | --- |
| **Vector Clock Timestamp** | **Description** | **Event Id** |
| [1,0,0] | Send msg AA to Mashi | a\_ashi |
| [2,0,0] | Compute “BBB” | b\_ashi |
| [3,0,0] | Send msg CC to Mashi | c\_ashi |
| [4,6,1] | Received msg "MM" from Mashi | d\_ashi |
| [5,6,1] | Send msg EE to Mashi | e\_ashi |
| [6,6,3] | Received msg "SS" from Sasi | f ashi |
| [7,7,3] | Received msg "NN" from Sasi | g\_ashi |

טבלה של משי

|  |  |  |
| --- | --- | --- |
| **Vector Clock Timestamp** | **Description** | **Event Id** |
| [0,1,0] | Compute “Start mashi” | h\_mashi |
| [1,2,0] | Received Msg AA from Ashi | i\_mashi |
| [1,3,1] | Received Msg QQ from Sasi | j\_mashi |
| [3,4,1] | Received Msg CC from Ashi | k\_mashi |
| [3,5,1] | Send msg LL to Sasi | l\_mashi |
| [3,6,1] | Send msg MM to Ashi | m\_mashi |
| [3,7,1] | Send msg NN to Ashi | n\_mashi |
| [5,8,1] | Received Msg EE from Ashi | o\_mashi |
| [5,9,6] | Received Msg VV from Sasi | p\_mashi |

טבלה של ששי

|  |  |  |
| --- | --- | --- |
| **Vector Clock Timestamp** | **Description** | **Event Id** |
| [0,0,1] | Send msg QQ to Mashi | q\_sasi |
| [0,0,2] | Compute “RR” | r\_sasi |
| [0,0,3] | Send msg SS to Ashi | s\_sasi |
| [3,5,4] | Received Msg LL from Mashi | t\_sasi |
| [3,5,5] | Compute “UU” | u\_sasi |
| [3,5,6] | Send msg VV to Mashi | v\_sasi |

סעיף ד':

להלן הקובץ המאוחד מסודר לפי ה causal order – כל האירועים המקביליים היו יוכלים להיות מסודרים אחרת, סידרתי אותם בצורה מסויימת אך זה גם היה יכול להיות אחרת.

|  |  |  |  |
| --- | --- | --- | --- |
| **Concurrent to** | **Vector Clock Timestamp** | **Description** | **Event Id** |
| H,Q,R,S | [1, 0, 0] | Send msg AA to Mashi | a\_ashi |
| A,B,C,Q,R,S | [0, 1, 0] | Compute “Start mashi” | h\_mashi |
| A,B,C,H,I | [0,0,1] | Send msg QQ to Mashi | q\_sasi |
| Q,R,S,H,I,J | [2, 0, 0] | Compute “BBB” | b\_ashi |
| A,B,C,Q,R,S | [1, 2, 0] | Received Msg AA from Ashi | i\_mashi |
| A, B, C, D, E, H, I, J, K, L, M, N, O | [0, 0, 2] | Compute “RR” | r\_sasi |
| Q,R,S,H,I,J | [3, 0, 0] | Send msg CC to Mashi | c\_ashi |
| A,B,C,Q,R,S | [1, 3, 1] | Received Msg QQ from Sasi | j\_mashi |
| A, B, C, D, E, H, I, J, K, L, M, N, O | [0, 0, 3] | Send msg SS to Ashi | s\_sasi |
| R,S | [3, 4, 1] | Received Msg CC from Ashi | k\_mashi |
| R,S | [3, 5, 1] | Send msg LL to Sasi | l\_mashi |
| R,S,T,U,V | [3, 6, 1] | Send msg MM to Ashi | m\_mashi |
| D,E,F,G,M,N,O | [3, 5, 4] | Received Msg LL from Mashi | t\_sasi |
| N,R,S,T,U,V | [4, 6, 1] | Received msg "MM" from Mashi | d\_ashi |
| D,E,F,R,S,T,U,V | [3, 7, 1] | Send msg NN to Ashi | n\_mashi |
| D,E,F,G,M,N,O | [3, 5, 5] | Compute “UU” | u\_sasi |
| N,R,S,T,U,V | [5, 6, 1] | Send msg EE to Mashi | e\_ashi |
| D,E,F,G,M,N,O | [3, 5, 6] | Send msg VV to Mashi | v\_sasi |
| N, O,P,T,U,V | [6, 6, 3] | Received msg "SS" from Sasi | f ashi |
| F,G,R,S,T,U,V | [5, 8, 1] | Received Msg EE from Ashi | o\_mashi |
| O,P,T,U,V | [7, 7, 3] | Received msg "NN" from Sasi | g\_ashi |
| F,G | [5, 9, 6] | Received Msg VV from Sasi | p\_mashi |