سوال ۱) الف) a) اگر یال back داشته باشیم به این معنا است که بین دو رأس این یال یکی نواده ی دیگری است. در BFS از جد به فرزندانش یک مسیر وجود دارد که حرکت رو به عقب ندارد. این یک تناقض است زیرا فرزندان هر رأس با جد خودشان تنها یک یال فاصله دارند یعنی هیچ مسیر دیگری به این فرزند وجود ندارد زیرا باعث می شود بیش از یک یال فاصله داشته باشد. به طور مثال برای یال forward داریم اگر همچین یالی در BFS وجود داشته باشد به این معنا است از یک رأس متوجه می شویم فرزندی دارد که قبلاً پردازش شده است که تناقض است زیرا هر فرزند یک یال با جدش فاصله دارد که یک بار توسط رأس دیگری کشف شده است.

b) یال (u,v) را در نظر بگیرید. در الگوریتم BFS اگر (u,v) یک یال درختی باشد به این معنا است که u پدر v می باشد و یک مسیر از u به v وجود دارد. اگر دقت کنیم این تنها زمانی اتفاق می افتد که $v.d = u.d + 1$ و چون تا پایان الگوریتم این وضعیت بدون تغییر باقی می ماند، اثبات کامل می شود.

c) لم: در طی اجرا الگوریتم BFS روی گراف G، صف Q شامل رئوس $\langle v_1, \dots, v_n \rangle$ است که v_1 ابتدا صف و v_n انتهای صف هستند. برای $i = 1, \dots, n$ داریم: $v_n.d \leq v_1.d + 1$ و یا $v_i.d \leq v_j.d$.

حال یال (u,v) را در نظر بگیرید که یک یال cross است. فرض کنیم u قبل از v وارد صف شده است. در زمان پردازش راس u راس v باید داخل صف باشد زیرا در غیر این صورت یال (u,v) انتخاب شده و رأس v از u کشف می شود و در نتیجه (u,v) یک یال tree است. پس از آنجایی که v داخل صف بوده بر اساس لم داریم $v.d \leq u.d + 1$ و بر اساس نتیجه گیری لم داریم $u.d \leq v.d$ اگر فقط تساوی هر دو عبارت را در نظر بگیریم اثبات تمام است.

ب) a) یال (u,v) را در نظر بگیرید. فرض می کنیم این یال یک یال forward است. هنگام پردازش رأس u این یال را پیمایش کرده ایم پس یک یال tree است.

b) مانند (الف-b) اثبات می شود. اگر (u,v) یک یال tree باشد یعنی یک مسیر از u به v است که در این مسیر u را اول می بینیم و u پدر v است پس فاصله ی v با u برابر ۱ است. که در واقع این فاصله همان یال (u,v) است پس $v.d = u.d + 1$.

c) برای هر یال (u,v) چه cross باشد چه نباشد نمی‌توانیم داشته باشیم $v.d > u.d+1$ زیرا هنگام بررسی (u,v) ، v در آخر کار کشف شده است پس $v.d \le u.d+1$.

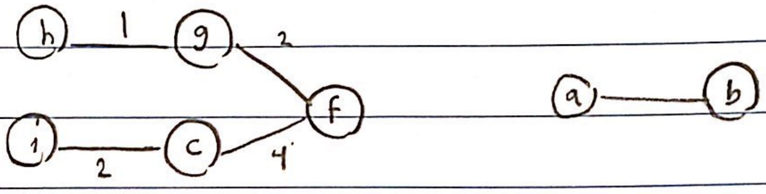
d) برای هر راسی مانند v داریم $v.d \ge 0$ ، اگر (u,v) یک یال back باشد ، راس v در جهت اول سطح ، والد u است پس $v.d \le u.d$ / حلقه‌ها را یال‌های back در نظر گرفتیم که می‌توانیم بگوییم $u=v$)

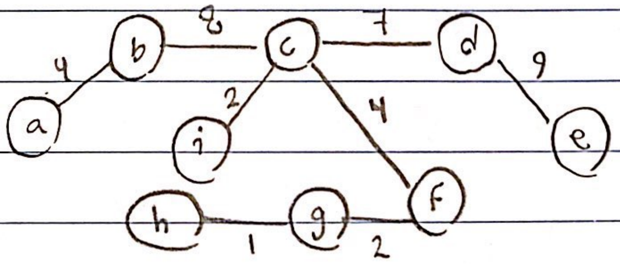
سوال 2) از پایین شبکه شروع می‌کنیم و هر راس را با شماره تعداد راه‌هایی که به آن راس می‌رسد مشخص می‌کنیم . برای هر راس جمع اعداد روی راس‌هایی که یال به سمت آن راس دارد را می‌نویسیم .
در اینجا فقط یک راه از راس s به راس F وجود دارد ، راه مجاور به s دو راه برای رسیدن به راس F دارد . به طور مشابه برای هر دو طرف s داریم . راه مجاور به s ، 3 راه برای رسیدن به راس F دارد برای آن یکی سمت هم همینگونه است . پس همه‌ی راه‌ها برابر است با $3+2+1+2+3=11$

پس 11 راه برای spider وجود دارد تا عنکبوت در شبکه به fly برسد .

سوال 3) ابتدا یال ها را سورت می کنیم به ترتیب صعودی ، حال به ترتیب هر یال و دو راسی را که وصل می کند را انتخاب کرده و یک disjoint-set برای آن ها تشکیل می دهیم و مطمئن می شویم دور تشکیل نشود

الف)





ب) می دانیم مرحله ی اول الگوریتم Kruskal سورت کردن یال ها می باشد حال وقتی اعداد صحیح هستند در یک بازه مشخص می توانیم با Counting sort و در زمان خطی سورت کنیم.

Time complexity Counting sort:

$O(\underset{\downarrow}{\underline{n}} + \underset{\downarrow}{\underline{k}}) \rightarrow O(E + |V|)$

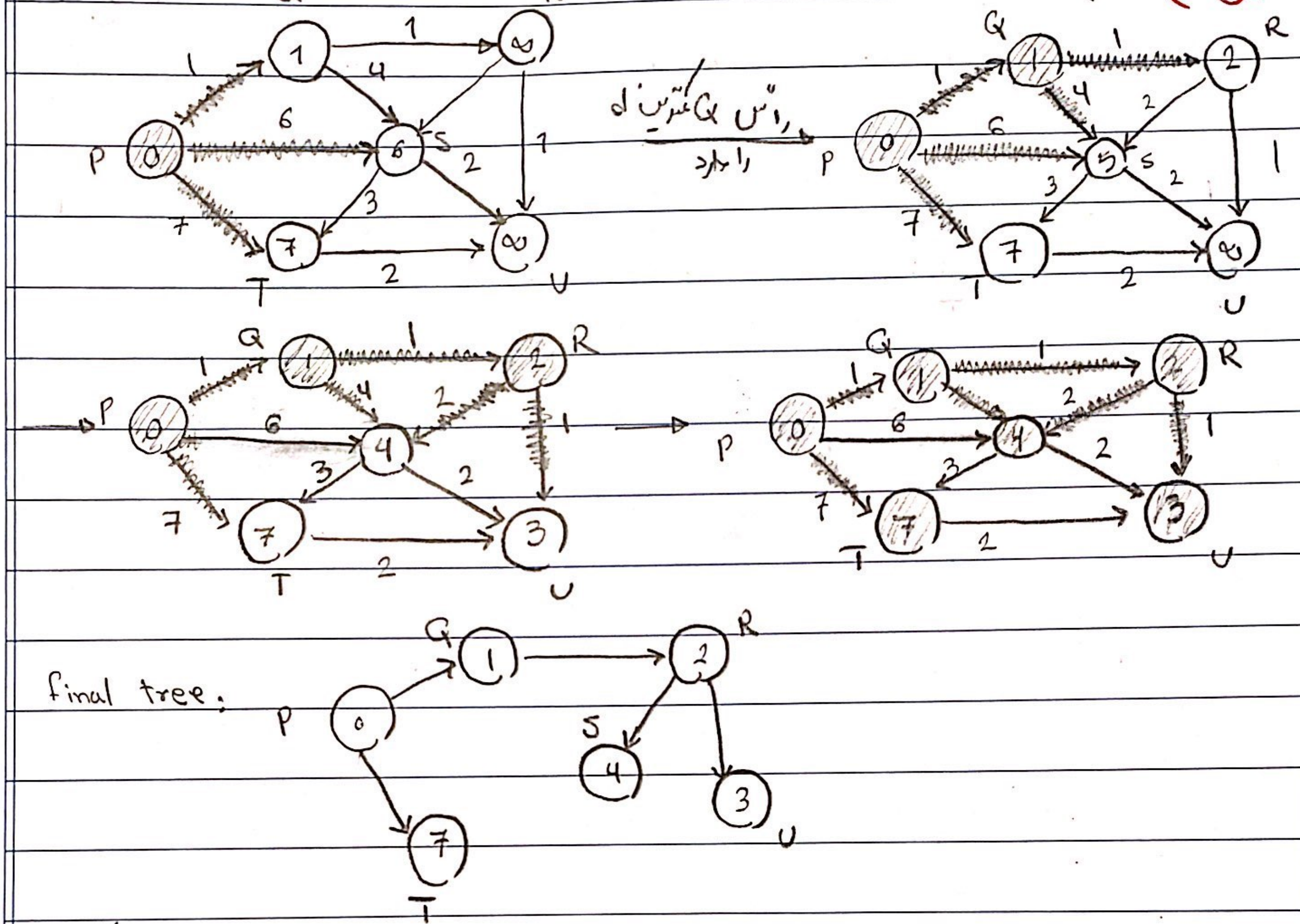
$\quad\; E \quad\;\; |V|$

$\rightarrow O(E + V + V\lg V) \Rightarrow O(E + V\lg V)$

Time Complexity Kruskal:

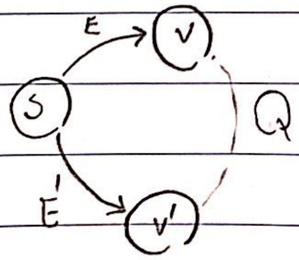
$O(E\lg V) \rightarrow O(V\lg V)$

سوال ۴) الف)



ب) با توجه به صورت سوال دور با وزن منفی نداریم و یال ها با وزن منفی به رأس مبدأ یعنی S وصل اند. حال باید ثابت کنیم به ازای هر $V \neq S$ که یال با وزن منفی به آن وارد می شود در محاسبه کوتاه ترین مسیر از S به V این یال با وزن منفی وجود داشته باشد. سپس می توانیم صحت کارکرد الگوریتم Dijkstra را مانند CLRS اثبات کرد. حال با برهان خلف اثبات می کنیم.

فرض کنید یال با وزن منفی E باشد و فرض کنید کوتاه ترین مسیر از S به V، $S \xrightarrow{E'} V' \xrightarrow{Q} V$ باشد



پس داریم: $E' + Q < E \Rightarrow E' + Q + E < 2E$

پس دور $E + Q + E'$ منفی می شود. بنابراین با فرض سوال که دور با وزن منفی نداریم در تناقض است.

سوال 5

$$D^{0} = \begin{bmatrix} 0 & 3 & \infty & 2 & 6 \\ 5 & 0 & 4 & 2 & \infty \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & 1 & 0 & 4 \\ 5 & \infty & \infty & \infty & 3 \end{bmatrix} \qquad \pi^{0} = \begin{bmatrix} NIL & 1 & NIL & 1 & 1 \\ 2 & NIL & 2 & 2 & NIL \\ NIL & NIL & NIL & 3 & NIL \\ NIL & NIL & 4 & NIL & 4 \\ 5 & NIL & NIL & NIL & 5 \end{bmatrix}$$

$$D^{1} = \begin{bmatrix} 0 & 3 & \infty & 2 & 6 \\ 5 & 0 & 4 & 2 & 11 \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & 1 & 0 & 4 \\ 5 & 8 & \infty & 7 & 0 \end{bmatrix} \qquad \pi^{1} = \begin{bmatrix} NIL & 1 & NIL & 1 & 1 \\ 2 & NIL & 2 & 2 & 1 \\ NIL & NIL & NIL & 3 & NIL \\ NIL & NIL & 4 & NIL & 4 \\ 5 & 1 & NIL & 1 & 1 \end{bmatrix}$$

$$D^{2} = \begin{bmatrix} 0 & 3 & 7 & 2 & 6 \\ 5 & 0 & 4 & 2 & 11 \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & 1 & 0 & 4 \\ 5 & \infty & 12 & 7 & 0 \end{bmatrix} \qquad \pi^{2} = \begin{bmatrix} NIL & 1 & 2 & 1 & 1 \\ 2 & NIL & 2 & 2 & 1 \\ NIL & NIL & NIL & 3 & NIL \\ NIL & NIL & 4 & NIL & 4 \\ 5 & 1 & 2 & 1 & NIL \end{bmatrix}$$

$$D^{3} = \begin{bmatrix} 0 & 3 & 7 & 2 & 6 \\ 5 & 0 & 4 & 2 & 11 \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & 1 & 0 & 4 \\ 5 & 8 & 12 & 7 & 0 \end{bmatrix} \qquad \pi^{3} = \begin{bmatrix} NIL & 1 & 2 & 1 & 1 \\ 2 & NIL & 2 & 2 & 1 \\ NIL & NIL & NIL & 3 & NIL \\ NIL & NIL & 4 & NIL & 4 \\ 5 & 1 & 2 & 1 & NIL \end{bmatrix}$$

$$D^{4} = \begin{bmatrix} 0 & 3 & 3 & 2 & 6 \\ 5 & 0 & 3 & 2 & 6 \\ \infty & \infty & 0 & 5 & 9 \\ \infty & \infty & 1 & 0 & 4 \\ 5 & 8 & 8 & 7 & 0 \end{bmatrix} \qquad \pi^{4} = \begin{bmatrix} NIL & 1 & 4 & 1 & 1 \\ 2 & NIL & 4 & 2 & 4 \\ NIL & NIL & NIL & 3 & 4 \\ NIL & NIL & 4 & NIL & 4 \\ 5 & 1 & 4 & 1 & NIL \end{bmatrix}$$

$$D^{5} = \begin{bmatrix} 0 & 3 & 3 & 2 & 6 \\ 5 & 0 & 3 & 2 & 6 \\ 14 & 17 & 0 & 5 & 9 \\ 9 & 12 & 1 & 0 & 4 \\ 5 & 8 & 8 & 7 & 0 \end{bmatrix} \qquad \pi^{5} = \begin{bmatrix} NIL & 1 & 4 & 1 & 1 \\ 2 & NIL & 4 & 2 & 4 \\ 5 & 5 & NIL & NIL & 4 \\ 5 & 5 & 4 & NIL & 4 \\ 5 & 1 & 4 & 1 & NIL \end{bmatrix}$$