DIJKSTRA(G,w,s)

```
INITIALIZE_SINGLE_SOURCE(G,s)
S = ∅
Q = G.V        ▷ inserting all vertices into Q
While Q ≠ ∅
      u = EXTRACT_MIN(Q)
      S = S ∪ {u}
      for each vertex v ∈ G.Adj[u]
            RELAX(u,v,w)
```

سوال ۵ - بخش الف)

Loop invariant ما :

در هر Iteration داریم :

for each $v \in S \rightarrow d[v] = \delta(s,v)$

پس کافی است نشان دهیم برای هر u قبل از اضافه کردن آن به S داریم:

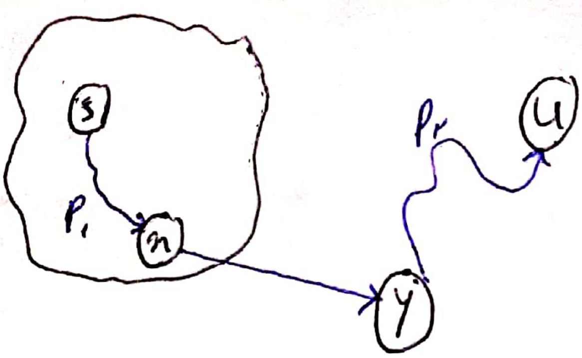
$$d[u] = \delta(s,u)$$

با تکیه بر upper-bound کار خود را شروع می کنیم :

Initialization ← در ابتدا $S = \emptyset$ قرار داریم پس loop invariant ما صحیح است.

Maintenance ← فرض می کنیم u اولین رأس از مجموعه V است که هنگام اضافه شدن به S ← $d[u] \neq \delta(s,u)$. داریم که $s \neq u$ زیرا s اولین رأس اضافه شده به مجموعه S است و $d[s] = \delta(s,s) = 0$. پس می دانیم که $S \neq \emptyset$. می دانیم که باید یک مسیر از s به u موجود باشد وگرنه $d[u] = \delta(s,u) = \infty$ که با فرض اولیه ما تناقض دارد. پس یک SP داریم که یک رأس به نام s از مجموعه S را به یک رأس به نام u از مجموعه V وصل می کند.

حال اگر فرض کنیم یک رأس مانند y در مسیر باشد که $y \in V$ می توان رأسی مانند x را به عنوان predecessor آن در مسیر P فرض کرد که $x \in S$.



یعنی چیزی شبیه به شکل روبرو ←

پس می توان نوشت که

$$P = s \overset{p_1}{\rightsquigarrow} x \rightarrow y \overset{p_2}{\rightsquigarrow} u$$

(p_1 و p_2 می توانند حاوی هیچ یالی نباشند) .

حال ادعا می کنیم که $d[y] = \delta(s,y)$

اثبات : می دانیم که u اولین حالت تناقض ما در loop invarient است پس $d[x]=\delta(s,x)$ ، همچنین هنگام اضافه کردن x به S همه edge های آن من جمله (x,y) ، relax شده است. حال با توجه به این که y زودتر از u در مسیر P ظاهر شده است و براساس فرض همه edge دارای وزن نامنفی هستند :

$$\star \quad d[y] = \delta(s,y) \le \delta(s,u) \le d[u]$$

ولی می دانیم که u زودتر از y هنگام extract_min انتخاب شده است پس $d[u] \le d[y]$

در نتیجه تنها حالت ممکن $d[u]=d[y]$ است.

با توجه به ← $d[y]=\delta(s,y)=\delta(s,u)=d[u]$ پس اثبات شد که هنگامی که u را به S اضافه می کنیم شرط $\delta(s,u)=d[u]$ برقرار است.

Termination : الگوریتم زمانی به پایان می رسد که $Q=\emptyset$ که در آن زمان داریم $S=V$

و با توجه به loop invarint ← for each vertex $v \in V \rightarrow d[v]=\delta(s,v)$

سؤال ۵ - بخش ب)

می توانیم دقیقا از ایدهٔ الگوریتم DIJKSTRA استفاده کنیم با چند تغییر :

Algorithm :

INITIALIZE_SINGLE_SOURCE(G,s) → در این قسمت به جای ∞ کردن d را برای همه راس ها ۰ می کنیم.

$S = \emptyset$

$Q = G.V$

while $Q \neq \emptyset$

  u = EXTRACT_MAX(Q) → به جای extract_min از extract_max استفاده می کنیم زیرا بیشترین مقدار d را می خواهیم.

  $S = S \cup \{u\}$

  for each vertex $v \in G.Adj[u]$

    RELAX(u, v, r)

چون مطمئن ترین راه بین ۲ راس را می خواهیم می توانیم یک شرط اضافه کنیم که هرگاه به آن راس رسید ، الگوریتم پایان یابد.

* RELAX(u,v,r) :

if $d[v] < d[u] \times r(u,v)$:

  $d[v] \leftarrow d[u] \times r(u,v)$

  $\pi[v] \leftarrow u$