Shortest Path Problem Assignment

Question 1
The LP is as follows:
max $\sum_i y_i * 1$
s.t: $\forall i \in [1, |S|] : y_i \geq 0$
and $\forall l \in [1, m] : \sum_{s:l\in\delta(s)} y_s \leq w_l$

Question 2
It can only be updated once. On each iteration the connected component gets shifted thus the C component can only be affected in one stage.

Question 3
We want to show that after the termination there will exist a path P from s to t in F.
On each iteration of the algorithm, the connected component F_k expands by one node not aready present.
Therefore, this means that in at most $n-1$ iterations all nodes will have been added.Hence, there needs to be a path from s to t, because the final F will be a tree with $n-1$ edges.
Of course, said path might occur earlier(and thus we might exit the algorithm earlier)

Question 4
Induction Proof
for $i = 1$:
Any graph with one edge is a tree. Furthermore, the edge that is added belongs to $\delta(\{s\})$ so it does contain s
Let it hold for $i = 1....n-1$
Let us now prove it holds for $i = n$ as well
We already know that F_n contains s (this is an inductive proof and F_{n-1} is a subgraph of F_n).
A new edge w is added to F_{n-1}. $w \in \delta(F_{i-1})$ so it doesnt connect two nodes already connected. Hence, F_n remains a Tree with n nodes and $n-1$ edges.

We will use the Weak Duality Principle.
$val(LP_{max}) \leq val(LP_{min})$
but LP_{max} is the dual.
$val(y^*) \leq val(L_{min})$
Furthermore, $val(L_{min}) \leq OPT$ since it is a minimization problem.
Thus, $val(y^*) \leq OPT$

Question 5
All $Y_i \geq 0$ because they start from 0 and can only be increased. Thus, if the dual solution is invalid then, there should be some $l : \sum_{s:l\in\delta(s)} y_s \geq w_l$. But this can never happen because we stop increasing once these two quantities are

equal. This leads to a paradox and our original assumption was wrong.
Therefore, in the 'worst case', these two quantities will be equal(when edge is added), as further increase for the first quantity is prohibited by the algorithm.

Question 6
From (5) we have that y is valid.
Furthermore, $val(y) \leq val(y^*)$ as it is a maximization problem.
Therefore, $val(y) \leq val(y^*) \leq OPT$

Question 7
Since $e \in P \Rightarrow e \in F$, it means that $\sum_{s:e\in\delta(s)} y_s = w_e$

Question 8
$\sum_{e:e\in P} w_j =$
$\sum_{e:e\in P}(\sum_{s:e\in\delta(s)} y_s)$

Question 9
From (Question 8): $\sum_{e:e\in P}(\sum_{s:e\in\delta(s)} y_s)$
I will shift the summations symbols:
$= \sum_{s:e\in\delta(s)} y_s \sum_{e:e\in P}(1)$
This means that for each s we add y_s for every e such that $e \in \delta(s)$ and $e\ inP$
In other words, when $e \in P \cap\ \delta(s)$.
Adding this in our equation, we get : $\sum_s y_s|P \cap\ \delta(s)|$

Question 10
Since $y_S > 0$ then at some point i our component was S.
i.e S is a part of F tree.
Since $|P \cap\ \delta(S)| \geq 2$ we have at at least two edges exit S.
Hence, there exist two different nodes (remember: it is path not cycle) a, b such that $a, \gamma, ..., \zeta, b$ where both $\gamma, \zeta \in \delta(S)$
But S is also a connected component, which means that there is a path in S connecting a, b.
As a result, there are two paths connecting a, b in F. The once induced by P and the one in S.
But F is a tree, which means that there is only one path between any nodes.
Our assumption was wrong and $|P \cap\ \delta(S)| = 1$

Question 11
$\sum_{e:e\in P} w_j = \sum_s y_s|P \cap\ \delta(s)|$ (Question 9)
$\sum_{e:e\in P} w_j \leq \sum_s y_s$
$\sum_{e:e\in P} w_j \leq OPT$ (Question 6)
Therefore, the algorithm is an 1-approximation algorithm.
Which means that the algorithm is optimal

Question 12 It is important to bound the factor of the sum and give the proof of optimality!

Question 13
Initially, all $y_z = 0$.

Djikstra:
The second node to i be added to D is the one with minimizes the $d(s, j)$ distance for all j which are neighbours of s. In Primal-Dual terms, the edge in $\delta(\{s\})$ with the minimum weight.

Primal Dual:
Only $y_{\{s\}}$ will start increasing until it 'hits' $w(e)$ where $e \in \delta(\{s\})$. Amongst all such e the one that will be added is the one that has the minimal weight(Otherwise, the dual constraint for some other edge with smaller weight would have become invalid).

We can see that the definitions in both paragraphs are the same. The edge will the minimal weight will processed in either algorithm.

Question 14
The node i which will be added is the one that minimizes $l(j)$ for all possible j. Let this edge be (a, i). (a, i) in $\delta(C_0)$ will be the first to become tight, which is exactly how the algorithm operates.

Question 15
All those edges (j, k) where $k \in V - S^{'}$.

Question 16
$l_{S'}(k) = minimum(l_S(k), l_S(j) + w(j, k))$

Essentially, we try to check whether the new edge (j, k) makes any difference. Since (j, i) has just become part of $\delta(S^{'})$ it means that all y in its dual constraint are 0. Hence the time needed for this particular constraint to become tight is $w(j, k)$. So overall time is previous time plus the new one.

Question 17
They are added in the same order. One can see that l has the same recurrence function as d. Furthermore, from Question 13 we know that they begin by processing the same edge ($l_1 = d_1$) (i.e they have the same initial conditions). From this we can conclude that indeed they are the same.

Question 18
There is the classic and easy to implement $O(mlogn)$ algorithm and a more advanced algorithm using Fibonacci heaps with complexity $O(m + nlogn)$ where n, m are the size of node and edges respectively.