assignment 2.

Q2.

(a) We use directed acyclic graph (DAG) here.
First, we assume S_0 be the set of nodes which is without prerequisite, remove all nodes in S_0 from $G_0 = G$ and reduce the in-degree of other nodes if there're incoming edges from the nodes in S_1.
Then let G_1 be the resulting graph and S_i be the nodes in G_i with in-degree zero. We assume $G_{k+1} = \emptyset$ and create a sequence of sets $S_0, S_1, S_2 \dots S_k$. with $k+1$ semesters to finish all the courses.

Next, we implement the algorithm. We construct an auxiliary graph $G'' = (V \cup \{v_0\}, E \cup \{(v_0, u)\})$. It is obvious that G'' is a DAG and takes $O(|V|+|E|)$ time. The algorithm begins to iterate until $S_{i+1} = \emptyset$, where $S_0 = \{v_0\}$, $i=0$, and i is the current iteration.

Thus, for each node $u \in S_i$, the algorithm traverses along the link list of node u, for $u \in Adj(u)$, the in-degree of node v is reduced by 1, if its in-degree is 0, v is added to S_{i+1}. When neighbours of each node in S_i have been tested, i is assignmented to $i+1$ (if $S_{i+1} = \emptyset$). Then the algorithm started to next interation until it terminates

We assume that there are k disjoint subsets $S_1, S_2, \dots S_k$, then $\bigcup_{i=1}^{k} S_i = V$. For each node $u \in V$ in these subsets,

the time cost of constructing subsets is:

$$\sum_{u \in \bigcup_{i=0}^{k} S_i} \text{out-degree}_{G'}(u) = O(|V|+|E|).$$

where out-degree$_{G'}(u)$ is the outgoing degree of node u in G''.

Thus, the algorithm takes $O(|V|+|E|)$ time.

(b)

Firstly, remove all edges from the algorithm we created whose reliability probability is 0 as such edges fail and can be ignored.

Let P be a most reliable path from s to t which contains edges (v_0, v_1), $(v_1, v_2) \cdots (v_{i-1}, v_i)$, (v_{k-1}, v_k) where $v_0 = s$, $v_k = t$. Then let $p_i = p(v_{i-1}, v_i)$. Thus, the reliability of P is $R(p)$, $R(p) = p_1 \cdot p_2 \cdots p_k$, which is maximized. (we assume)

Thus, $\frac{1}{R(p)}$ is minimized. and $\log \frac{1}{R(p)} = -\log p_1 - \log p_2 - \cdots - \log p_k$ is also minimized., where $-\log p_i \geq 0$, for i in $1 \leq i \leq k$, $0 \leq p_i \leq 1$.

Therefore, for the network $G=(V, E)$, we assume for each edge (u, v) in it, the edge weight is $w(u,v) = -\log p(u,v)$. That is, to find the most reliable path from s to t, we need to find a shortest path from s to t with edge weight w.

We now use Dijkstra's algorithm. Let L be the length of a shortest path from s to t. There exists two situations:

① L is infinite, there is no directed path from s to t.

② L is finite, the reliability of the most reliable path from s to t is 2^{-L}.

And the time complexity of the algorithm is:

$$O(|E|\log|V| + |V|\log|V|) = O(|E|\log|V|).$$

where Min-heap is applied.

Q3 (b).

We assume that the string is $X_1, X_2, X_3 \cdots X_n$. and create an array l to record the locations of cut, where $0 < l_1 < l_2 < l_3 \cdots < l_n < n$.

Then we assume the minimum cost of breaking a string $x_i x_j$ at one of the locations $C_{i+1} < C_{j-1}$ is $C(i,j)$.

where $C(i, i+1) = 0$ for all $0 < i < n$.

Thus, the solution for the question is $C(1, n)$.

Thus, we have the recursion:

$$C(i,j) = \min_{i<k<j} \{C(i,k) + C(k,j) + (C_j, C_i)\}.$$

We have $O(n^2)$ pairs of (i,j) and for each (i,j), it takes $O(n)$ time to traverse from i to j. Thus, the time complexity is $O(n^3)$.