Qn.34(34-1)

a)
Decision problem:
Is there some independent set of (sizes at least K) given a graph and a number k?(1)

If we take the compliment of the given graph, then it will have a clique of size at least k if and only if the original graph has an independent set of sizes at least k.
which means subset <= original set
Since if we take any set of vertices in the original graph, then it will be an independent set if and only if there are no edges between those vertices.

> In order to prove that the independent set problem is NP-hard, the polynomial time reduction of an instance of a Clique problem should be transformed in such a manner that the true input or instance of a clique problem's should be mapped with a true instance of an independent set.
>
> • Consider the instance of the Clique problem as $G(V,E),x$. This problem is independent set problem having set $G'(V,E'),x$ where the symbol E' is complement of E.
>
> • The set of vertices V represents a clique having size *x* is in the graph G only in a case if V' is an independent set of graph G' having the size of *x*, and it also following the fact that the construction of G',b from the G,b must be done in polynomial time.
>
> In this regard, it can be concluded that independent set problem is also NP-hard. Thus, as it has been proved that an independent set problem is NP as well as NP-Hard so it can be said that it is always **NP-complete,** which are the hardest problems in NP.

However, in the compliment graph, this means that between every one of those vertices, there is an edge, which means they form a clique.
So, to decide independent set, just decide clique in the compliment.
Then we can say that decision problem (1) is NP-complete.

b)
given a black-box subroutine to solve the decision problem define in part(a) and a graph G(V,E). here, it is asked to give an algorithm which can be used to find a independent set of max size. We set the black-box a B(G,x)

Algorithm:

// perform search operation

Step1: Start binary search on *B* to determine the maximum size x^* for the independent set.

// initialization step

Step2: Set the independent set I to be an empty set.

Step3: for each $v \in V$

Construct $G'(V',E')$ by removal of v and its related edges from the graph $G(V,E)$

Step 4: if ($B(G',x^*)$

Set $G \leftarrow G'$

else

$G \leftarrow G''$

Here, G'' is obtained by removal of all the vertices which are connected to *v* and the edges which are linked to it from the graph G.

// end

Here, Step 1 has time complexity of $\mathrm{O}(\log V)$.

Step2: It has the complexity of $\mathrm{O}(1)$.

Step-3, 4: They have the number of iterations equal to $(\mathrm{O}(E) \times \mathrm{O}(V))$

hence, the vertex in G which is obtained at last is independent set of sizes X* by the construction.
Hence, it can be concluded that the time complexity is O(V+E).

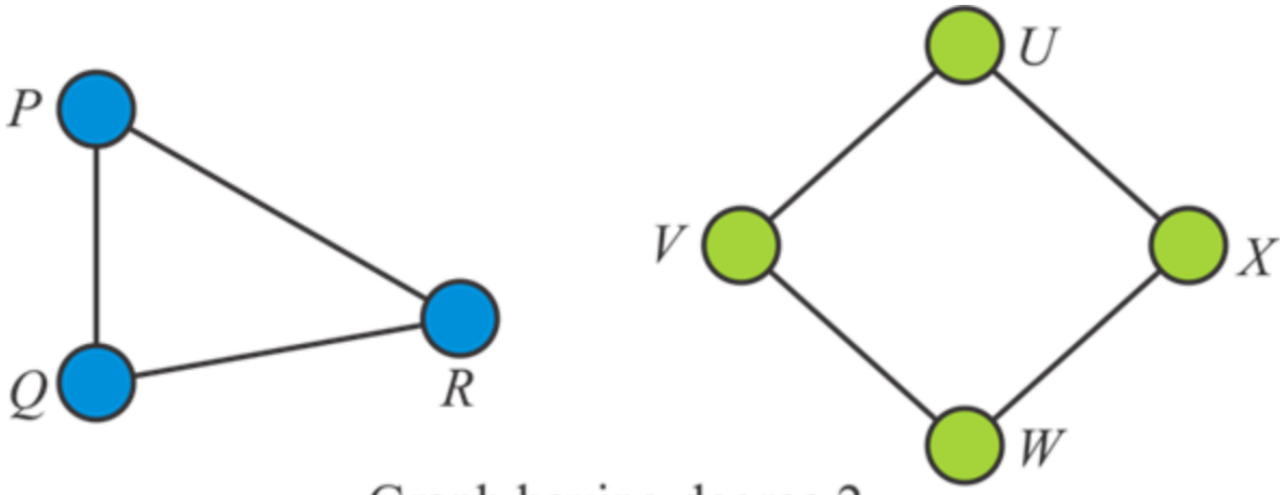
c)

Given that the degree of the graph *G* is 2, which implies that each vertex in the graph has a cardinality or degree 2.

Here, it is required to **give an algorithm that can solve the independent set problem of a graph having degree 2.** Also, it is required to analyze the running time and the correctness of an algorithm.

As, it can be noticed that when a degree of the graph is 2 then, in order to solve the problem of an independent set we must have the graph as a simple cycle.

Consider the graphs given below of degree 2 and thus, containing a simple cycle.



Graph having degree 2

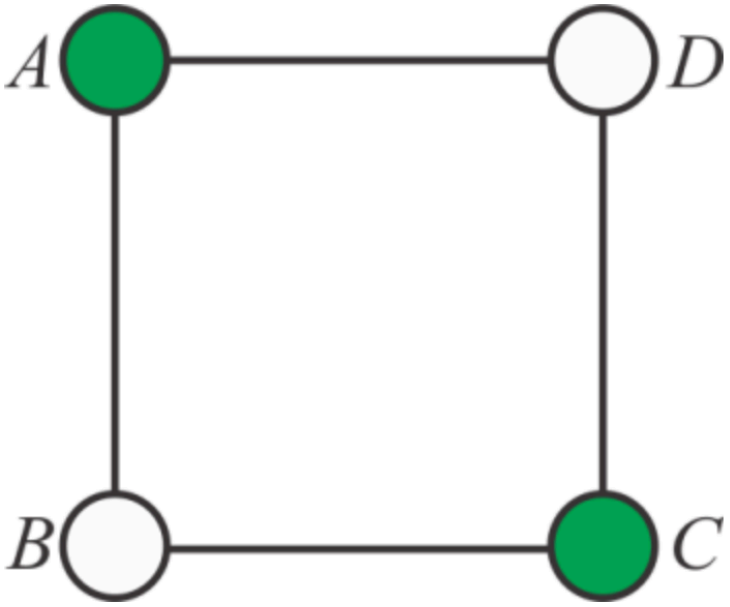
• Thus, from the above graphs it can be observed that generally the graph containing the vertices of degree 2 is a simple cycle.

• Therefore, the independent set problem in such a case can be achieved by initiating at any vertex and start choosing the alternate vertex on the cycle till the size obtained for the independent set to be $\lfloor |V|/2 \rfloor$.

Hence, it can also be concluded that the running time of the algorithm to solve the problem of independent set having *V* vertices, *E* edges and degree of each vertex as 2 is $O(|V|) or O(|E|)$.

To prove the correctness of the algorithm:

Consider the graph of degree 2 given below having vertices *A*, *B*, *C*, *D* and the set of four edges connecting them:



Start the process from vertex *A*. Now, according to the algorithm, proceed the traversal by picking up an alternate vertex and thus, vertex *C* would be chosen, leaving behind the vertex *B*. After choosing the vertex *C*, vertex *D* will not be picked and the algorithm stops with independent set containing the vertices $\{A, C\}$ and having size $\lfloor |V|/2 \rfloor$.

Hence, the solution obtained by applying the algorithm on the graph of degree 2 is a maximum independent set of size 2 which is $\lfloor |V|/2 \rfloor$.

Thus, the algorithm devised above is accurate and it will work correctly.

d)
First, find a maximal matching. This can be done in time O(V E) by the methods of section 26.3. let f(x) be defined for all vertices that were matched, and let it evaluate to the point that is paired with x in the maximal matching. Then, we do the following procedure. Let S1 be the unpaired points, Let S2 = f(N(S1)) \S1, where we extend f to sets of vertices by just letting it be the set containing all the pairs of the given points. Similarly, define Si+1 = f(N(Si)) \($U^{i}_{j=1}$ Sj).
First, we need to show that this is well defined. That means that we want to make sure that we always have that every neighbor of Si is paired with something. Since we could get from an unpaired point to something in Si by taking a path that is alternating from being an edge in the matching and an edge not in the matching, starting with one that was not, if we could get to an unpaired point from Si , that last edge could be tacked onto this path, and it would become an augmenting path, contradicting maximalist of the original matching. Next, we can note that we never have an element in some Si adjacent to an element in some Sj .
Suppose there were, then we could take the path from an unpaired vertex to a vertex in Si, add the edge to the element in Sj and then take the path from there to an unpaired vertex. This form an augmenting path, which would again contradict maximalist. The process of computing the {Si} must eventually terminate by becoming Φ because they are selected to be **disjoint** and there are only finitely many vertices. Any vertices that are neither in an Si or adjacent to one consist entirely of a perfect matching, that has no edges going to picked vertices. This means that the best we can do is to just pick everything from one side of the remaining vertices. This whole procedure of picking vertices takes time at most O(E), since we consider going along each edge only twice. This brings the total runtime to O(V E).