***def MyVC(G=<V,E>):***

***v = some vertex of G***

***tree T = run Prim's algorithm on G using v as source and 1 as weight of any edge /\* runs in polynomial time \*/***

***/\* the vertices of T are exactly those in V***

***return the set of non-leaf nodes of T***

***Let OPT denote the size of the optimal vertex cover of G. Let APPROXVC denote the set returned by the heuristic and APPROX denote its size.***

***(a). [2 point] Show that APPROXVC forms a vertex cover.***

PRIMS algorithm traverses all vertices for preparing the **MST**, hence all vertices of input G are included in T. Now since all vertices are included in the tree T, for any edge **(u,v) in G** , both **u & v** are present in T. Hence, T forms a vertex cover of G.

The above method is returning **APPROXVC** as the non-leaf nodes of **T**. Trimming the leaf nodes does not affect its property of being a vertex cover of G.

Say, there was an edge (x,y) in G such that x is an internal node & y is a leaf node. Even if y is removed from the set of vertices, still x is present in the set to cover that edge.

***Hence, removing all leaf nodes is an invariant for the vertex cover property, provided that there are no edges in G between the leaf nodes in T.***

***(b). [1 point] Show that any vertex cover of T (and hence, OPTT) can be modified to contain no leaf node.***

For any edge (x,y) in T such that x is an internal node & y is a leaf node, 2 cases can occur in the vertex cover of T:

* *Both x & y are present in the vertex cover :*

In this case, we can remove y, as x would still cover that edge.

* *Either of x or y are present in the vertex cover :*

In this case, we can always replace y with x, so that the edge (x,y) is still covered.

**Thus,** any vertex cover of T can be modified to contain no leaf nodes.

So is the case with **OPTT,** it can also be modified to contain no leaf nodes by the above operations.

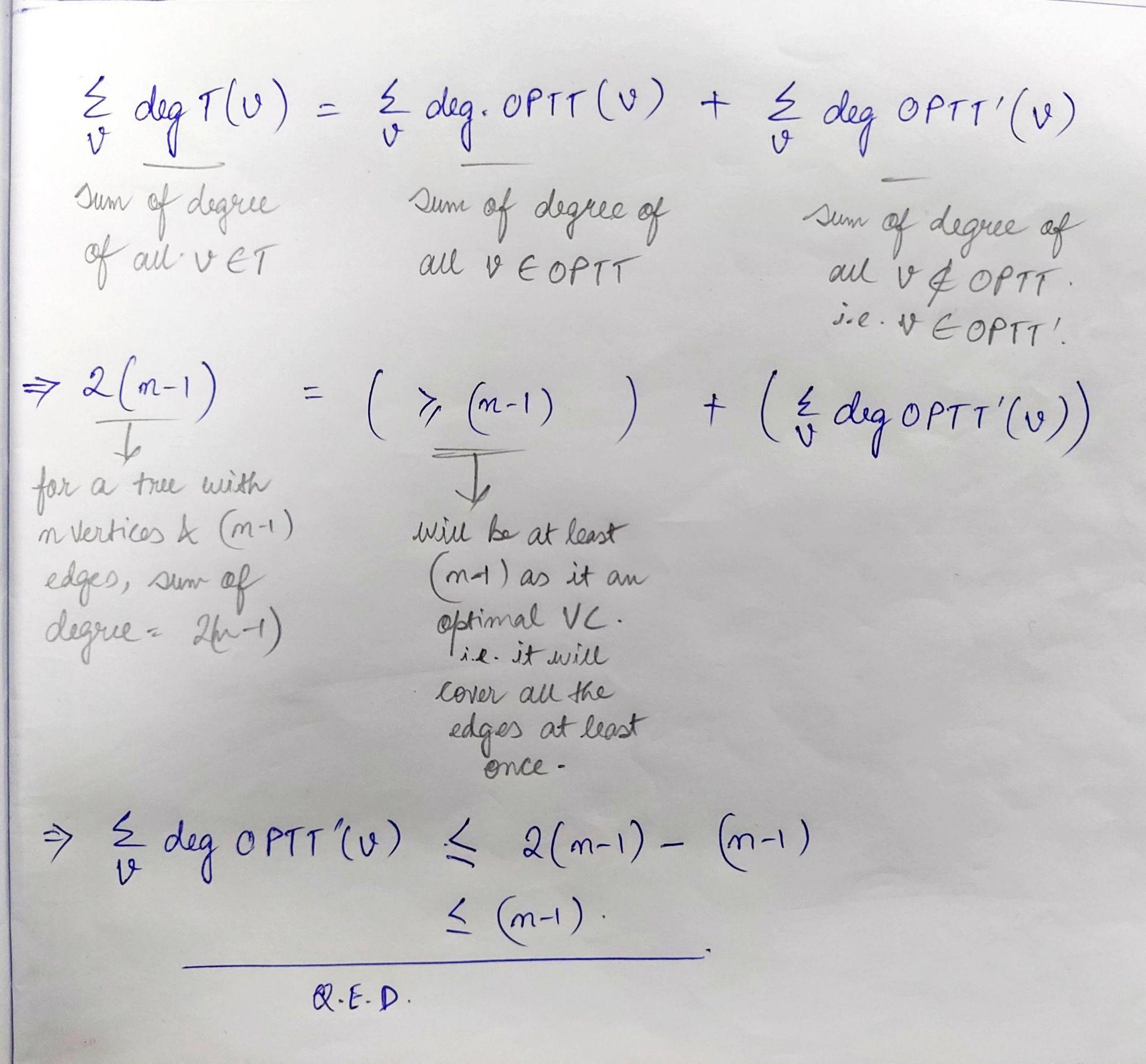
***(c). [1 point] Show that J = n - L - |OPTT|***

Let OPTT be any optimal vertex cover of T. Let L be the number of leaf nodes of T. Let

J be the number of internal nodes of T that are not in OPTT.

Given the fact that OPTT does not contain any leaf nodes; so any node **v** which is neither a leaf node, nor belongs to OPTT will definitely belong to J **i.e.** **v** must be an internal node which is not in OPTT. This also conforms to the definition of **J.**

***(d). [2 point] Show that sum\_{v not in OPTT} degT(v) <= (n-1)***

****

***(e). [1 point] Show that sum\_{v : internal node not in OPTT} degT(v) >= 2J***

Given that J represents the number of internal nodes of T that are not in OPTT. Each internal node of a tree must have at least 2 edges, one each with its parent & child.

Hence, the sum of degrees of all nodes in J will be at least 2J.

***(f). [1 point] Show that (n-1) >= L + 2J***

Since n = J + L + |OPTT| , thus any node which will not be in OPTT will be either in J or in L.

So, sum\_{v not in OPTT} degT(v) = sum\_{v in J} degT(v) + sum\_{v in L} degT(v)

= ( >= 2J ) [ from ***(e)*** ] + ( = L ) [ as leaf nodes

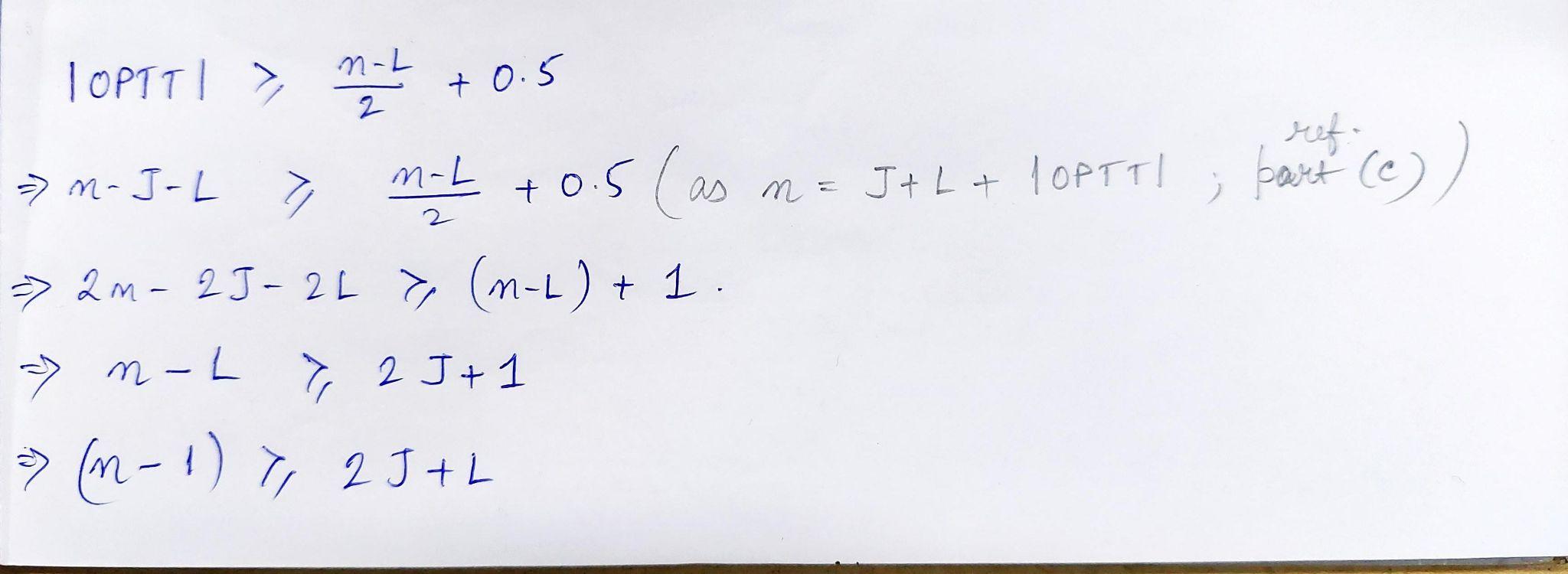
have one edge only]

Thus, **sum\_{v not in OPTT} degT(v) >= (2J + L)**

Also from ***(d),*** it is known that **sum\_{v not in OPTT} degT(v) <= (n-1)**

**Hence, (n-1) >= L + 2J**

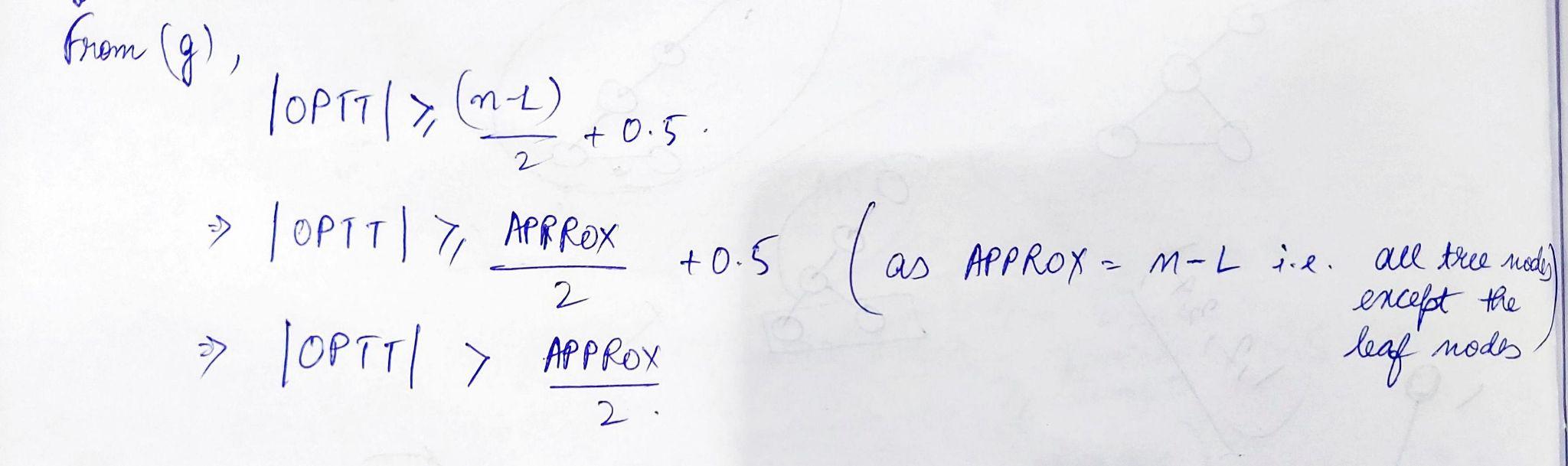
***(g). [1 point] Show that |OPTT| >= (n-L)/2 + 0.5***



I have already proved this result in part ***(f).***

Thus, **|OPTT| >= (n-L)/2 + 0.5** also holds true.

***(h). [1 point] Show that OPT > APPROX/2. Then compute the approximation ratio of the above algorithm?***



Since **OPT** is a vertex cover of the original graph G, so it will cover at least all the tree edges & some more non-tree edges also, if any.

Hence, **OPT >= OPTT. …(1)**

And, **OPTT > APPROX/2 …(2)**

**From (1) & (2) : OPT > APPROX/2**

**=> 2\*OPT > APPROX …(3)**

And since, APPROX is only a good vertex cover of the MST T**(& not the optimal vertex cover of graph G),**

**Hence, APPROX >= OPT …(4)**

**From (3) & (4) : 2\*OPT > APPROX >= OPT**

**Hence, the approximation ratio will be 2 i.e. the above algorithm is 2-relative Vertex Cover approximation algorithm.**