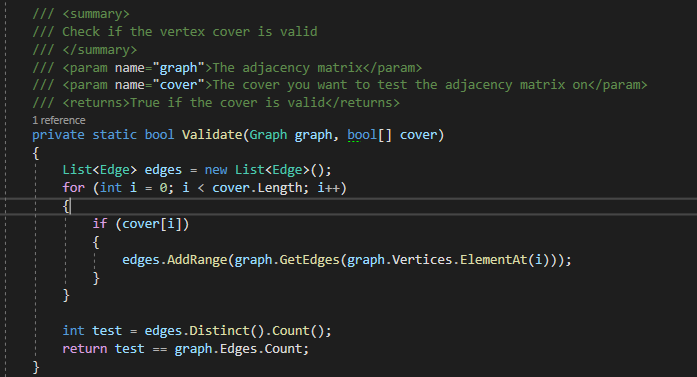
Week 2

# Finding the vertex cover

For the vertex cover we created a recursive method that checks if the vertex cover is possible. This method works by passing our custom graph class. We pass a reference to our cover so we can find the generated cover. What the method does is first check if the request size is the current size of the cover. If this is the case we check if our result is valid.

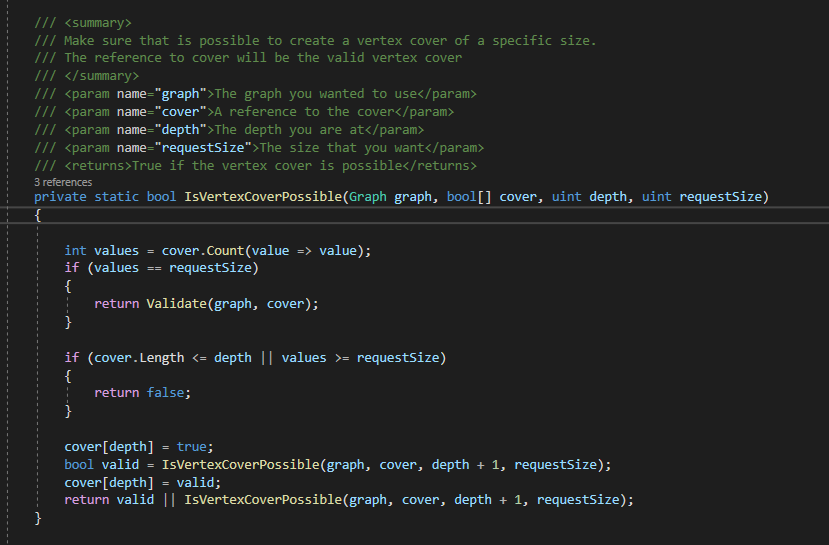
We do this with this method:

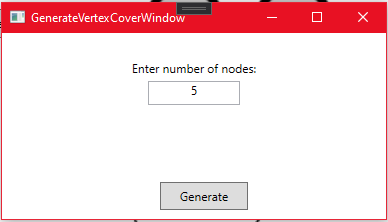


It works by checking whether all edges adjacent to the vertices in the cover are all the actual edges in the graph.

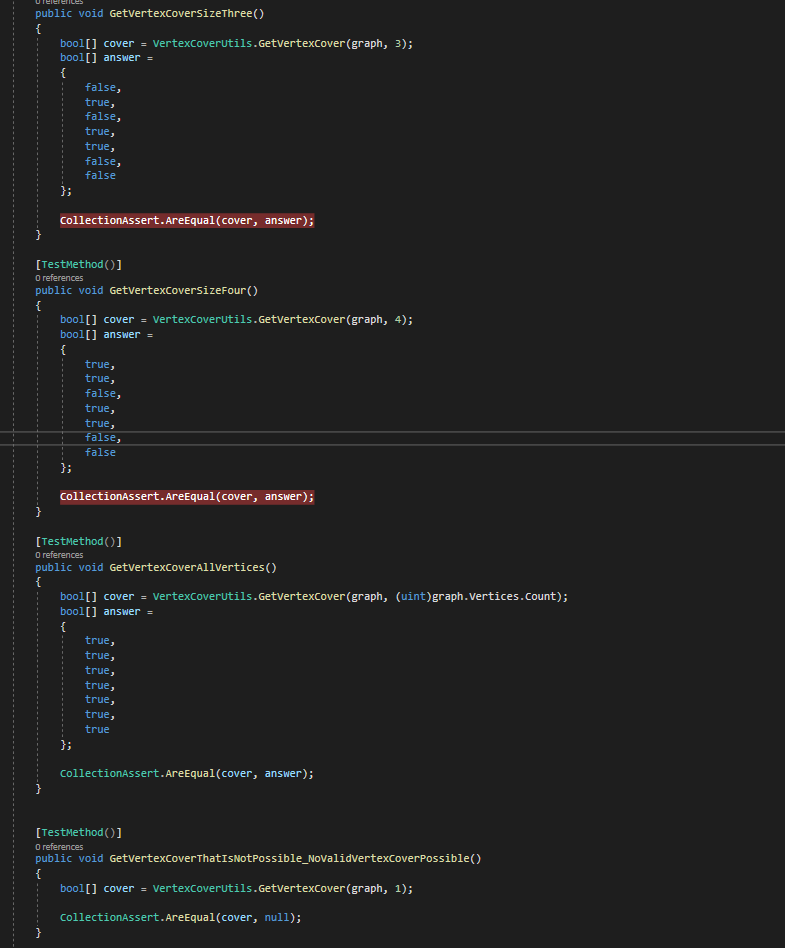
Then we check wether this cover is even possible we do this by checking wether the values currently inside the cover are larger than the requestsize. Also if the depth is above the size of the vertex cover we return false.

Otherwise we ecreate two paths one where the current vertex is covered in the vertex cover and the other is when it is not covered. So our method first search the lowest possible values. This looks like this:



To get the size of the cover we added the small pop up window:  


# Testing



# 

# 

# Graph viz with attributes

After finding the requested size of the vertex cover we wanted to add a way so that we can easily have attributes for every edge and vertex. We created a class that has a dictionary and all references to a specific graph. We then updated our graph viz system to incorporate the values. So we where able to color our graphs like this:

