# Biological Computation
# Assignment 1

Himanshu Arora

June 2, 2025

## Explanation of code

The main work flow in Q1 is :

1. Generate all connected graphs (not digraphs) with n nodes.
2. Then I get the isomorphic equivalence classes from the previous step.
3. The graphs from the previous step are used to generate digraphs. If there is an edge $(u, v)$ is in the original graph, then there must exit 3 graphs, such that there is an edge from $u$ to $v$ in one graph, an edge in the opposite direction in the second graph and in the third graph, both edges exist in the third graph.

   $\forall G, u, v.\exists G_1, G_2, G_3.(u, v) \in G \rightarrow (u, v) \in G_1 \wedge (v, u) \in G_2 \wedge (u, v) \in G_3 \wedge (v, u) \in G_3$.
4. The final step is to generate equivalence classes of graphs based on isomorphism.
   (a) I use the Weisfeiler Lehman Graph hash to generate a hash value for each graph. This hash value has the property that if 2 graphs have different hash values they are guaranteed to be non-isomorphic. But in case their has value is identical there is a strong probability that they are isomorphic.
   (b) I use the hash value above to make buckets. Then within each bucket I check for isomorphism to confirm.
   (c) This hashing technique reduced the number of isomorphic tests from quadratic to linear.

## Declaration

I did not use chat gpt to write the code but I took suggestion to design the workflow and chatGpt mentioned Weisfeiler Lehman Graph hash algorithm.

# Question 1

b)
```
n= 1
count= 0
```

Figure 1: Q1, output for $n = 1$

```
n= 2
count= 2
# 1
0 1
# 2
0 1
1 0
```

Figure 2: Q1, output for $n = 2$

For $n = 3$ and $n = 4$, please see `3.out` and `4.out`.

c) My implementation terminates for `n = 4` in an hour.

d) `n = 4` is the maximum termination that I have witnessed. For `n = 5` the program does not terminate within 10 hours.

# Question 2

I generate all the sub graphs of the given graph and then I find the equivalence classes over the set of graphs based on isomorphism.