Biological computation:

Code explanation:

- After inputting the desirable size in the txt file “size.txt”, it reads the size and sends it to the function “totalGraphs”, and then sends it to the function “subGraph”. The function subGraph creates a list and using function “sGraph” to find all subgraphs.
- The function “sGraph” recursively finds all subgraphs. If the size is 1 or 2 the function immediately outputs all possible subgraphs, and if the size is larger it recursively goes the smaller sizes until it reaches base case size = 2.
- Function combi uses subgraphs of size = size-1 and finds all possible edges of a new node (of size=size) to the nodes in the smaller subgraphs, and so finds all possible combinations of edges in graph. It adds the new edges to the smaller subgraphs and so finds newer subgraphs.
  In summary- using subgraphs of smaller size (by 1) it adds all possible combinations of edges with the new node and so finds all new subgraphs of the real size.
- After that, list of all the possible subgraphs return to the function “subgraphs” and from there it removes all isomorphism in a function called “removeIso”.
- Function removeIso goes over all the subgraphs in the list and compares them if it’s different graph and same size. If there is isomorphism it removes one of them.
- The list of the subgraphs without isomorphism return to “totalGraphs” and creates a new list of all the edges in each subgraph and sends it to “codeToText”.
- Function codeToText creates a text file and prints all the motifs according to the form written in the exercise.

main design and implementation decisions you made:

- We wanted to use a recursive function in order to easily find the subgraphs of each size. It is easier to use subgraphs of smaller size and add edges instead of each time finding all the subgraphs in a big size.
- We decided to use a real directional graph using the library network in order to have an easier way to remove isomorphism and also that the representation of the graph will be easier to understand (easily see all the edges and nodes).

1.

a) In order to run the code on a certain size create a txt file called ‘size’ and write ‘size=n’ where n I the desired size and run the code (like in the git depository).
   Output will come out as a txt file under the name “results_n=size_a” and print to the terminal.

b) The output of the results are in the repository.

c) 1,2,3,4 can finish within an hour, 5 can’t.

2.

- After inputting the desirable graph and size in the txt file “graph.txt”, it reads the size and sends it to the function “totalGraphs”. It also reads the edges of the graph and creates a directed graph with those edges in the code. From total graphs it sends the size to the function “subGraph”. The function subGraph creates a list and using function “sGraph” to find all subgraphs.
- The function “sGraph” recursively finds all subgraphs. If the size is 1 or 2 the function immediately outputs all possible subgraphs, and if the size is larger it recursively goes the smaller sizes until it reaches base case size = 2.
- Function combi uses subgraphs of size = size-1 and finds all possible edges of a new node (of size=size) to the nodes in the smaller subgraphs, and so finds all possible combinations of edges in graph. It adds the new edges to the smaller subgraphs and so finds newer subgraphs.
  In summary- using subgraphs of smaller size (by 1) it adds all possible combinations of edges with the new node and so finds all new subgraphs of the real size.
  We will later use this to find the motifs.
- After that, list of all the possible subgraphs return to the function “subgraphs” and from there it removes all isomorphism in a function called “removeIso”.
- Function removeIso goes over all the subgraphs in the list and compares them if it’s different graph and same size. If there is isomorphism it removes one of them. Now that we removed the isomorphism we have a list of all the motifs.
- The list of the motifs return to “totalGraphs”.
- We send the graph that was inputted and the size and send it to “sGraph2”. The function goes over the graph and find all edges, find all combinations of the edges, number of edges according to size, then create subgraphs of the combinations, find if all edges are connected and if they are add the subgraph to a list. So it outputs a list of all subgraphs in the desired size.
- We send the list of all motifs and the list of subgraphs and send it to the function “countMotif”. It goes over the motifs and check how many times they appear in subgraphs of the given graph and outputs a list with the number of appearances of each motif.
- We create a new list that has all the edges of each graph in the motif list and send that with size and list of countMotif to the function “codeToText”.
- Function codeToText creates a text file and prints all the motifs according to the form written in the exercise with the number of appearances they had in the inputted graph.

In order to run the code with a specific graph and size create a txt file called ‘graph’ and write ‘size=n’ where n I the desired size, and then in new lines write each edge that graph has. Run the code (like in the git depository).

Output will come out as a txt file under the name “results_n=size_b” and print to the terminal.