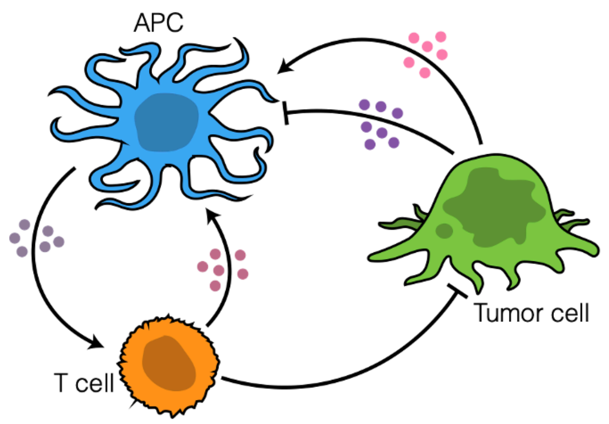
The *fil-rouge* project

Objectives: conciliate *bio* and *informatics*



The *computer-science* part:

- Graph theory and graph manipulation
- Object-oriented programming
- Data-science analysis (plotting, visualization)
- Programming

The *bio* part:

- Epidemiological modeling
- Regulation of biological networks
- Cellular Automaton
- Biological cycles

Our Expectations

We expect from you that you:

- will think about a system questions/topics
- will setup some simulations that could help answersing questions
- will take conlusions/decisions from the simulations
- will make a critical examination of the validity of your simulation model

We expect your code:

- to be correct, to produce a result
- to be commented
- **to be explained by each member of the team**

Teams

Teams of 4 (max two groups of 3 allowed! do not even try asking).

You must register your team on **Moodle before June 1st**.

Final Deliverables (1/2)

Implementation

- Python source code

Slides

- Detailed description of the problem
- Description of the simulation: *model*, *parameters*, *what does it try show ?*
- Description and analysis of the results with *plots* and *schemas*
- Conclusion and perspectives

Final Deliverables (2/2)

3. A live lecture / defense (in late June on Teams)

- 15’ presentation + demo
- 5’ discussion
    - we might ask you to show and explain your code

To be submitted on Moodle before the defense

Grade

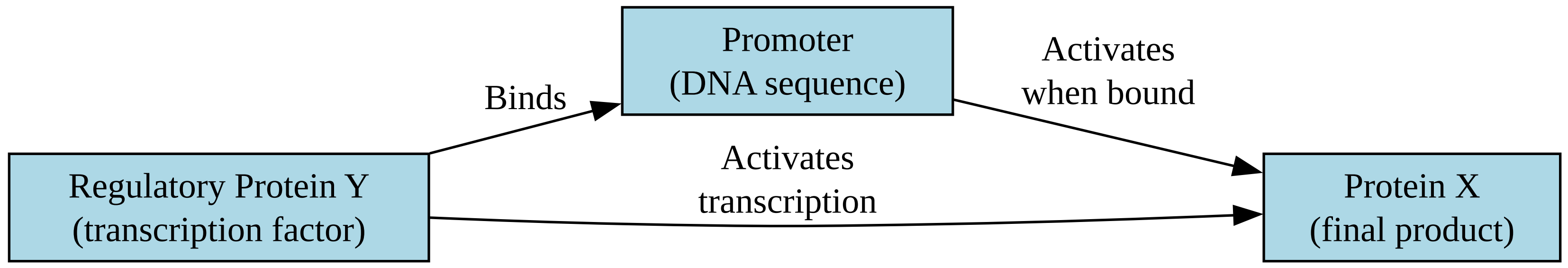
An evaluation grid will be provided to you.

Subject

Part 1: Gene[r|t]ic Circuit Simulation (code base)

Let's consider a genetic circuit involved in regulating the production of a protein X. This genetic circuit could comprise three main components:

- *Promoter*: The promoter is a DNA sequence to which a transcription factor binds, a protein that controls the activation or deactivation of gene transcription. In this circuit, we'll have a gene X whose transcription is regulated by a promoter.
- *Regulatory Protein Y*: Protein Y is a transcription factor that binds to the promoter of gene X and regulates its expression. Protein Y can act as an activator or a repressor of X transcription depending on its state.
- *Protein X*: Protein X is the final product of the gene regulated by this circuit.



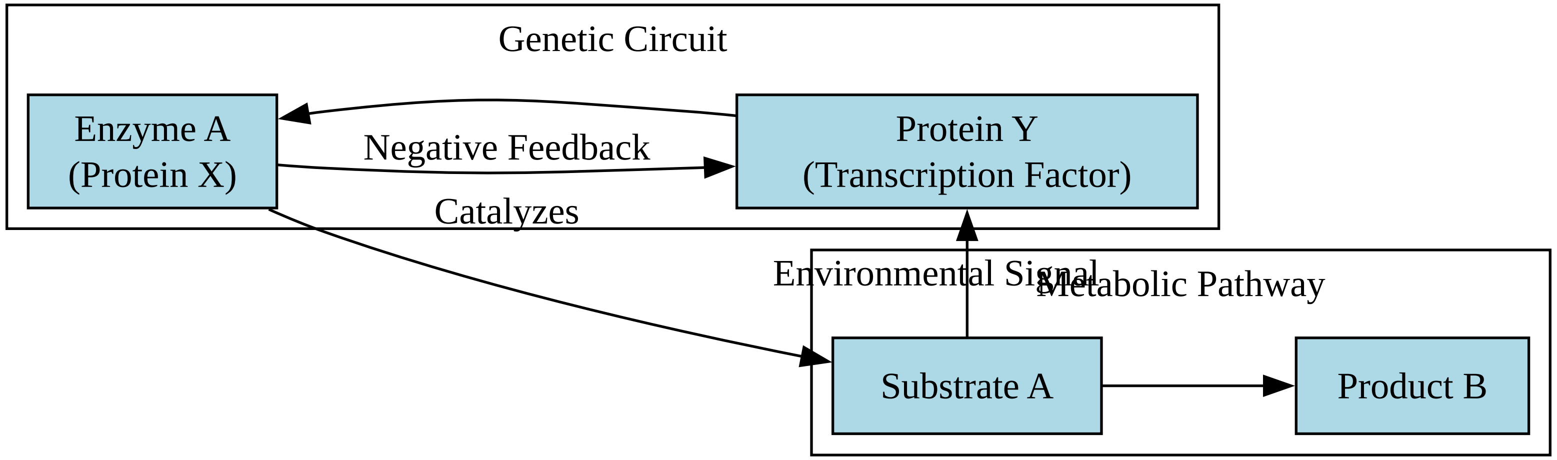
When protein Y is present and active (for example, based on an external signal), it binds to the promoter of gene X and activates X transcription. In this case, protein X is produced.

When protein Y is absent or inactive, it cannot bind to the promoter of gene X, which inhibits X transcription. In this case, the production of protein X is reduced or stopped.

This simple genetic circuit regulates the production of protein X in response to external signals or other environmental conditions.

More concrete example

- Protein X = **Enzyme A**. This is an enzyme involved in the metabolism of Substrate A. Enzyme A catalyzes a specific chemical reaction that converts *Substrate A* → *Product B*.
- Protein Y = **Transcription Factor** This protein acts as a transcription factor that binds to the promoter of the gene encoding Enzyme A.
- Regulatory Signals: the availability of Substrate A. If the concentration of Substrate A in the environment is high, this activates the production of Protein Y.



Interactions

Substrate Quantity (Substrate A)

> If [Substrate A] > activation_threshold: production of Y is activated.
> If [Substrate A] < inhibition_threshold: production of Y is inhibited.

Biochemical Reactions

Enzyme A catalyzes the chemical reaction: Substrate A $\rightarrow$ Product B. The rate of this reaction depends on the concentration of available Enzyme A and Substrate A.

> Rate = k * [Enzyme A] * [Substrate A], where k is the rate constant.

Gene Expression Regulation

> If Y is active: production of Enzyme A is activated.
> If Y is inactive: production of Enzyme A is inhibited.

Negative Feedback

The production of Enzyme A (Protein X) can also act as negative feedback on the production of Protein Y. Once the concentration of Enzyme A reaches a sufficiently high level, it can inhibit further production of Protein Y, thus regulating the expression of the gene encoding Enzyme A.

> If [Enzyme A] > feedback_threshold: production of Y is inhibited.

Versatile system simulation

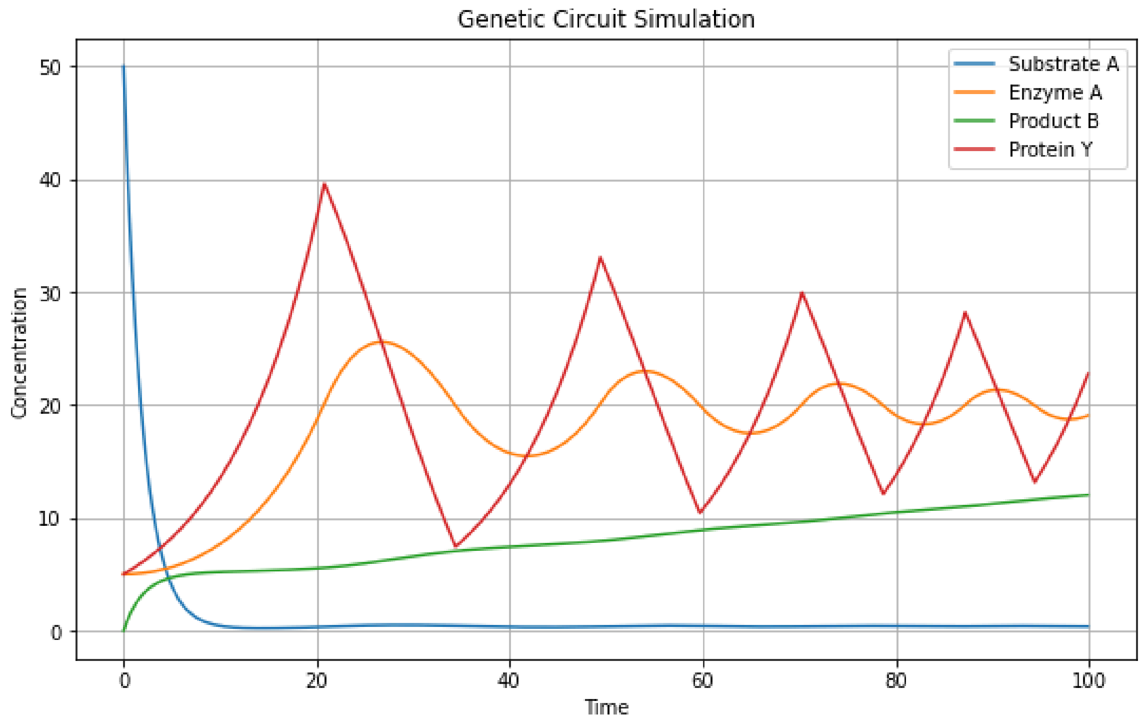
You have to use Object-Oriented Programming to model this system. It should be versatile enough to allow you to explore different scenarios and conditions.

```
circuit = Circuit()
circuit.add("Enzyme A", EnzymeA(quantity=10))
circuit.add("Protein Y", ProteinY(quantity=10))
circuit.add("Substrate A", SubstrateA(quantity=100))
circuit.add("Product B", ProductB(quantity=0))
circuit.simulate(steps=100)
```

> 💡 You still don't know about Object-Oriented Programming? Don't worry, we'll cover this in the next lectures.

Sample output

Simulate the system under different conditions and analyze the results. *Here we see the negative feedback of Enzyme A on the production of Protein Y*. You can explore the effects of changing the initial concentrations of the components, the rate constants of the reactions, and the thresholds for activation and inhibition.



Part 2: your custom system

You have a working genetic circuit simulation. Now, you want to extend this simulation to model a more complex system involving a circuit of **your choice**.

You can choose to model a different genetic circuit, a metabolic pathway, a signaling network, or any other biological system that interests you with more or less complex interactions.

In the presentation, you should be able to:

1. Describe the system you are modeling and the interactions between its components. A drawing of the interaction graph: nodes and activation and/or inhibition arrows should be included.
2. Determine the parameters and initial conditions of the system. Explain how these parameters affect the behavior of the system and how they make sense biologically. Explain the biological relevance of the system you are modeling (e.g., why the activator is stronger than the inhibitor etc.)
3. Analyze the system. Determine the number of states for each variable in the system. The number of states is the number of different levels or thresholds that a variable can take. For example, an *activation level* with two thresholds can take three states: below the lower threshold, between the two thresholds, and above the upper threshold.

> Create a table that summarizes the states for each *node* ?

4. Simulate the system under different conditions and analyze the results. You should be able to explain the behavior of the system under different conditions and how it responds to changes in the environment or the initial conditions. You can use plots, diagrams, or other visualizations to illustrate the behavior of the system. It might be interesting to also plot the "state" of each element of the system over time (not just the quantity).