**Readme file**

In this exercise we implemented a generic system for running different simulations.

In this exercise the interaction with user is with graphic interface.

We receive from the users the path of an xml file and from it we read all the data about the "world". The world contains entities on which the simulation is performed, with each entity having different characteristics that will be tested during the simulation. In addition, the world contains environmental variables that will affect in one way or another the entire course of the simulation.

We chose to divide the project into 3 modules:

1. systemEngine (contains all the logical implementation of the system), knows the dto module.

2. ui-java-fx (the communication with the user), knows the dto module and the systemEngine module.

3. dto (contains objects for transferring information between the ui and systemEngine), does not know any of the modules.

A brief explanation of the systemEngine module hierarchy:

- Contains an interface that wraps the entire logical part, and through which you can receive information (dto objects) and perform operations in the system. An implementation contains members for defining a world, and a list of world instances, which will be created during each simulation run.

- The world contains separate classes for defining the entities, properties and environment variables, and additional classes that will have instances of them only while running a simulation that are instances of all of these.

- In addition, there are departments that are responsible for creating dto's according to the ui request.

- There is a department whose role is to run a simulation, and in which you will create the world instance and everything it includes.

A brief explanation of the ui module hierarchy:

- Contains the Main class, in which there is a call to the method of the Ui class, which contains an instance of the systemEngine interface and which runs the main loop that offers the user his options, and from which there is a call to the methods of classes that realize the desired operation.

A brief explanation of the hierarchy of the dto module:

- Contains new definitions of the objects of the world (entity, property, etc.) that contain only the information we want to expose and get methods only.

- Contains various dtos that are adapted to the type of information desired to be transferred (from the ui to systemEngine and vice versa).

