Backend Task

Design a framework for building state-machines. The framework should provide as much 'common' functionality as possible, without imposing limits on concrete implementations and allowing simple use.

Components:

State: An abstract class. derived classes implement the logic of the state machine and should know how to handle different types of events and indicate transitions to a new state

Event: Should have some identification, additional data... may be an abstract class (concrete implementations may receive different types of events)

Machine: should be able to receive events and move between states

Hint: When using the framework to define a specific machine, most work is done by implementing specific states and events. The machine component is probably not very different form one machine to another.

Assignment 1:

- Implement the components
- For testing, implement the following state machine use case using the framework you built: assume there are two types of events that are passed to the machine in our system. The machine should indicate (e.g. print a warning to the standard output) the first time it receives 3 consecutive events of the same type.

Assignment 2:

Make the machine persistent - save its "state" to a file and read it when the process starts so it can continue from where it left off. Assume that there will be more types of events/states in the future.

Language: Java/C#/C++/Python

Documentation: Nothing fancy - just add comments for the non-trivial parts

We recommend a recap on finite state machines.