Exercises module 24 – Enums

24.1 – Traffic light

In this exercise we are going to implement the logic to control a traffic light and implement the necessary state-machine to control this.

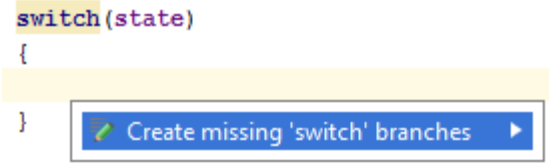
1. First we need to read more about what a state-machine is by reading:
   https://sv.wikipedia.org/wiki/Tillst%C3%A5ndsmaskin (Tillståndsmaskin)
   https://en.wikipedia.org/wiki/Finite-state_machine
   Google for "finite state machine java enum"
2. Our job is to implement the necessary logic to control a traffic light. First make a drawing on paper to sketch out the necessary states needed. Use an **enum** to represent the states.
3. One other requirement is that we need to implement this specific interface:

```
public interface TrafficLight {
    // Move the traffic light one step forward
    void Tick();
    boolean GetGreenLight();
    boolean GetYellowLight();
    boolean GetRedLight();
}
```

   Once started the traffic light will advance one step forward each time the Tick method is called and the traffic light system will call the Get methods to determine if a lamp should be visible or not.
4. We need to be compliant with the European standard pattern for the lights according to:
   (https://sv.wikipedia.org/wiki/Trafiksignal)

	S1	S2	S3	S4
Red	X	X		
Yellow		X		X
Green			X	

5. Implement the necessary logic to drive the lights using enum to keep the current state.
6. Use a **switch** statements to implement the state-machine. A useful trick is to let IntelliJ generate the necessary case statements by pressing **alt+enter** inside a switch statement:



7. You can use this code for the main method to verify that it works:

```
while(true)
{
    TLS.Tick();

    System.out.println("\r\n\r\n");
    System.out.println("Red: " + TLS.GetRedLight());
    System.out.println("Yellow: " + TLS.GetYellowLight());
    System.out.println("Green: " + TLS.GetGreenLight());

    Scanner sc = new Scanner(System.in);
    sc.next();
}
```

If you have time:

Some ideas for further improvements:

1. Try to make a more advanced traffic light, with different timing for each state?
   (the traffic light class returns the time for the main application to wait between each step, using a GetWaitTime method?)
2. Emergency mode where emergency vehicle can force the lights to be in a certain state until they release it, like using a EnterEmergencyMode(Color), ExitEmergencyMode()
3. Error mode
   Resulting in blinking yellow lights when there's some internal traffic light error
4. Think about how you would model an entire intersection with several lights and directions.