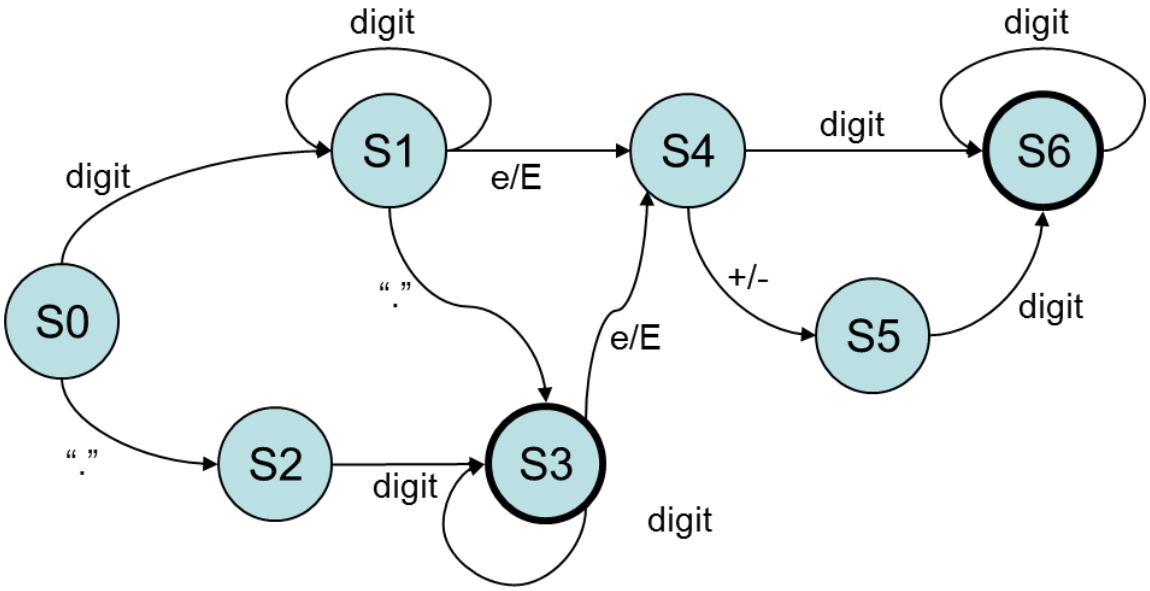
Worksheet: State Pattern

A typical application of the state pattern is the implementation of a state machine. With this worksheet you will implement a state machine for parsing floating point numbers. A floating point number is a number which contains a decimal point and/or a scientific extension. Examples of valid floating point numbers are:

```
1.33   0.4e10   .3   .4E+5   4e-3
```

The grammar of such numbers can be described with the following state diagram. *S0* is the start state and *S3* and *S6* are accepting end states. The transitions to the error state for other characters entered are not shown.



In project 04_State you find a program which implements this state diagram. In the provided code the state machine is implemented in a loop with one big `switch` statement.

```
State s = State.S0;
int pos = 0;
while(s != State.ERROR && pos < str.length()) {
    char ch = str.charAt(pos++);
    switch(s) {
        case S0:
            if(isDigit(ch)) { m = ch - '0'; s = State.S1; }
            else if(ch == '.') { s = State.S2; }
            else s = State.ERROR;
            break;
        case S1:
            ...
            break;
        case S2:
            ...
            break;
        ...
    }
}
```

It is your task to change this implementation and to make use of the state pattern.

First think about which methods must be contained in the State interface. Also think about how in each state the successor state is determined and set as the new current state.