

Week 5 assignments

When creating the program code, you must apply the following basic principles:

- create a separate project for each assignment;
- use name 'assignment1', 'assignment2', etcetera for the projects;
- create one solution for each week containing the projects for that week;
- make sure the output of your programs are the same as the given screenshots;

CodeGrade auto checks

Make sure all CodeGrade auto checks pass (10/10) for your assignments. The manual check will be done by the practical teacher.

Auto checks assignment 1-¹⁰/₁₀ **AT**

Auto checks assignment 2-¹⁰/₁₀ **AT**

Manual check

Automatic checks for assignment 1		
0	10	
	10	
	100	%

Submit

6.67

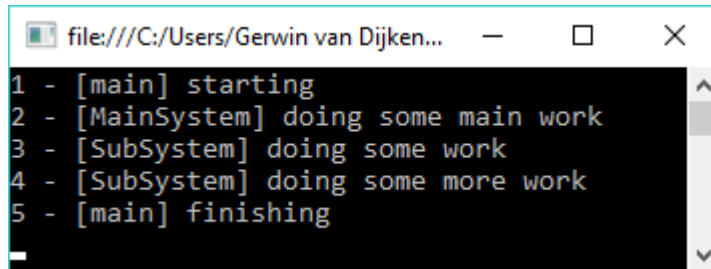
20 / 30

✖

⌵

Assignment 1 ('Singleton Pattern')

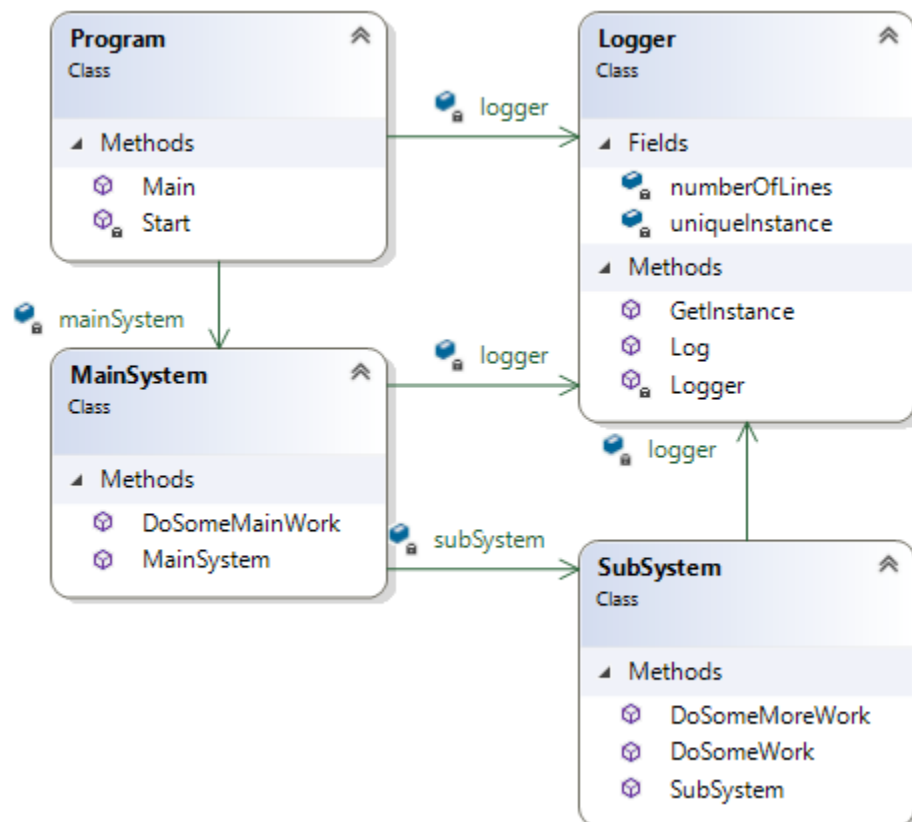
In an application logging is done at several places (in several classes) using a special Logger class. To save resources this Logger is implemented as a 'Singleton'. Each line in the log is preceded by a line number, starting at 1 and continuously increased by 1 when the next line is written (see screenshot below).



```
file:///C:/Users/Gerwin van Dijken...
1 - [main] starting
2 - [MainSystem] doing some main work
3 - [SubSystem] doing some work
4 - [SubSystem] doing some more work
5 - [main] finishing
```

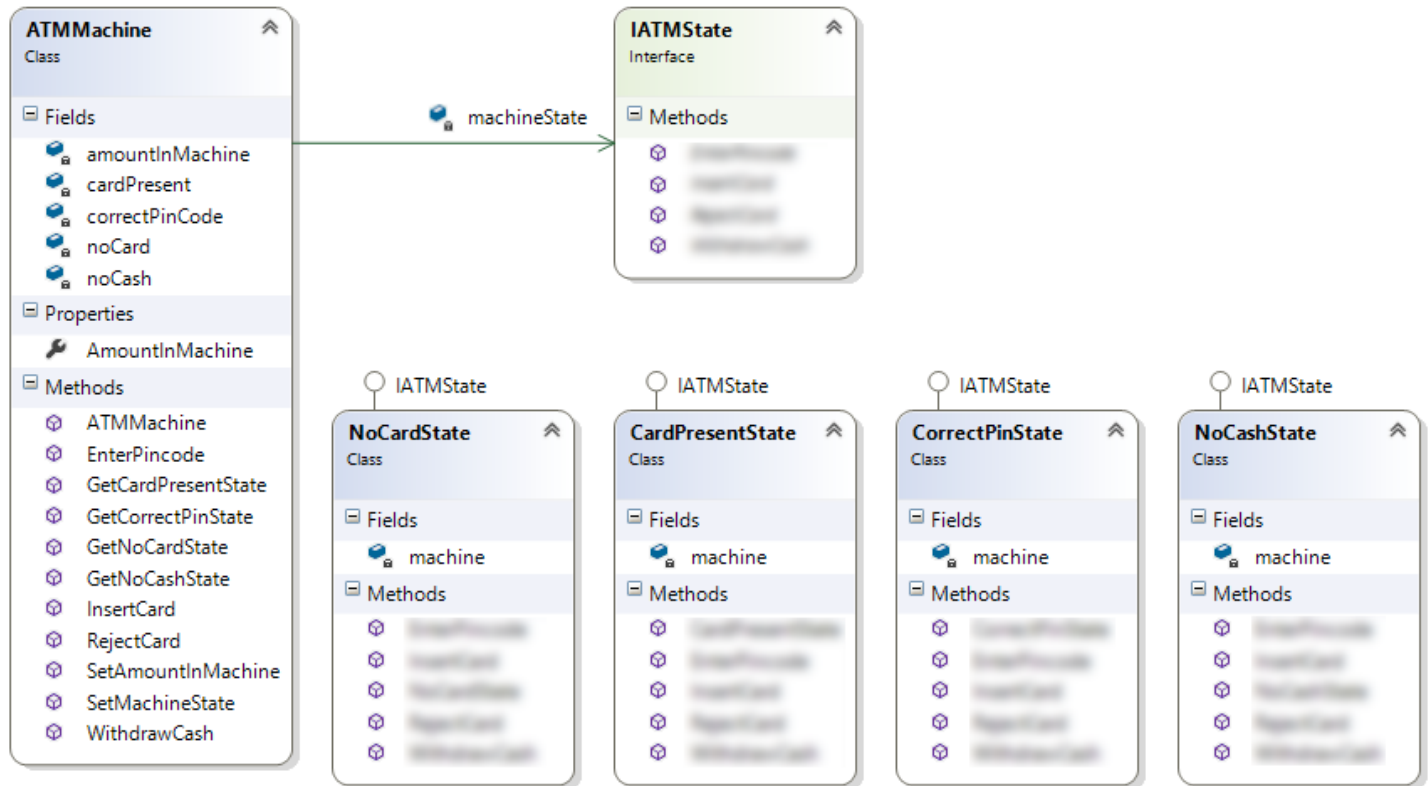
Implement the Logger class, a MainSystem class, a SubSystem class, and a simple Start method that (together) generates the output shown above. The Start method calls the logger with: `logger.Log("main", "starting");` and creates the MainSystem object. Do not pass the logger object around!

In the classdiagram you can see how the classes relate to each other.



Assignment 2 ('State Pattern')

Create an application with an ATM machine (cash dispenser) that has 4 different states: 1) no card present, 2) card present, 3) entered correct pincode and 4) no cash available. Use the 'state pattern' to implement the ATM machine and its states. Use the classdiagram below to implement the interface and classes.



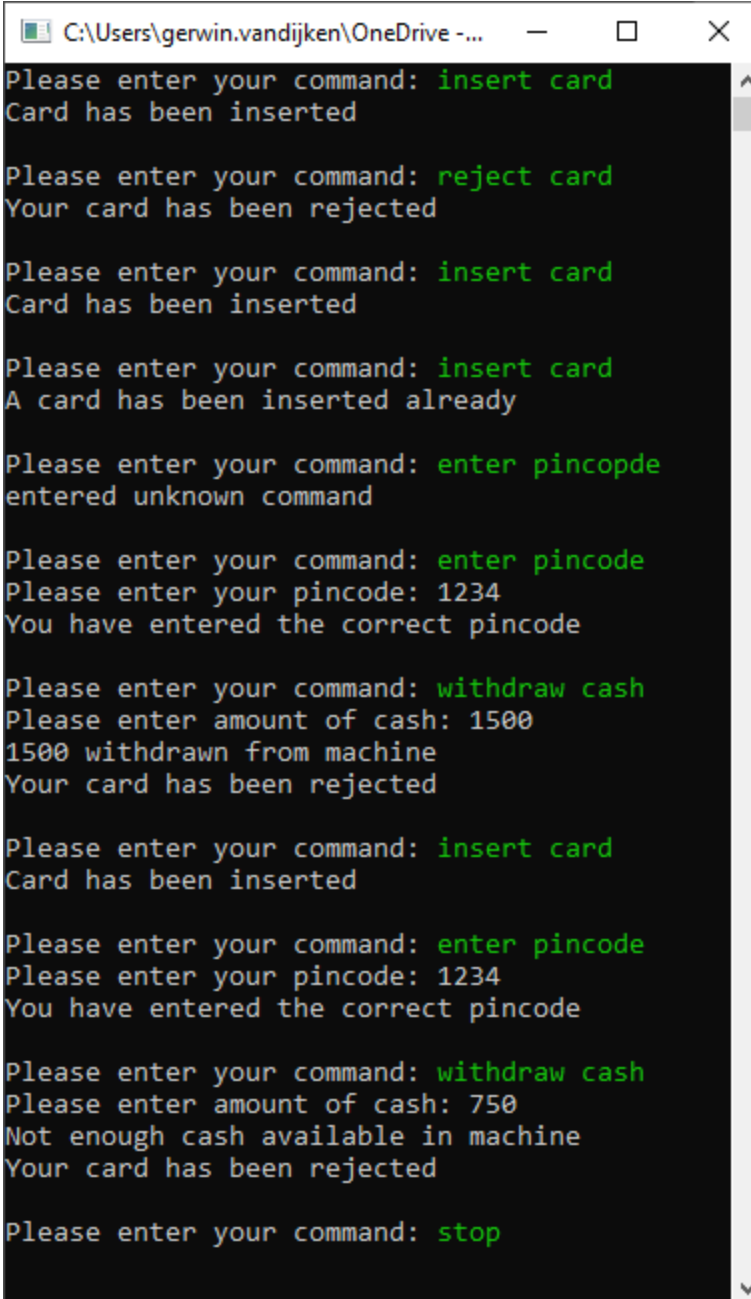
In the Start method, create an ATM machine object containing 2000 euro, and make a loop for reading a command (valid commands are: "insert card", "reject card", "enter pincode", "withdraw cash"). When the user enters "stop", the program ends.

```

void Start() {
    ATMMachine machine = new ATMMachine(2000);

    // loop...
}
  
```

You can see an example of the output on the next page.



```
C:\Users\gerwin.vandijken\OneDrive - ...  
Please enter your command: insert card  
Card has been inserted  
  
Please enter your command: reject card  
Your card has been rejected  
  
Please enter your command: insert card  
Card has been inserted  
  
Please enter your command: insert card  
A card has been inserted already  
  
Please enter your command: enter pincopde  
entered unknown command  
  
Please enter your command: enter pincode  
Please enter your pincode: 1234  
You have entered the correct pincode  
  
Please enter your command: withdraw cash  
Please enter amount of cash: 1500  
1500 withdrawn from machine  
Your card has been rejected  
  
Please enter your command: insert card  
Card has been inserted  
  
Please enter your command: enter pincode  
Please enter your pincode: 1234  
You have entered the correct pincode  
  
Please enter your command: withdraw cash  
Please enter amount of cash: 750  
Not enough cash available in machine  
Your card has been rejected  
  
Please enter your command: stop
```