This week the main activity was a quiz activity, with a structure similar to our Friday lecture activities. The retrospective for the quiz is in Quiz-07-retrospective.pdf This retrospective explores the proficiency point exercise that you had 24 hours after the end of your recitation to complete.

TASK

```
package code;

public class SavingsAccount {

	/* YOUR TASK - DUE 24 HOURS AFTER THE END OF YOUR RECITATION.
	*
	* Define this class to that represents a simple Savings Account.
	* Pay attention to the names, parameters and return types of the methods
	* described.  They must be exactly as given, else the code won't compile
	* on AutoLab.
	*
	* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
	*
	* The balance of the account must never go below zero.  A withdrawal must
	* not be done unless there is prior authorization.  If authorization has
	* been obtained a single withdrawal can be performed.  Performing an
	* authorized withdrawal rescinds the withdrawal, preventing any additional
	* withdrawals from taking place until authorization for another withdrawal
	* is obtained.
	*
	* The constructor of the class must initialize the balance to zero, and the
	* withdrawal authorization to false.
	*
	* The 'deposit' method must verify that the amount to be deposited is
	* non-negative before proceeding with the deposit.  The 'deposit' method
	* must have a void return type.
	*
	* The 'withdraw' method must verify that the amount to be withdrawn is
	* non-negative, that the amount is no greater than the current balance, and
	* that the withdrawal is authorized before proceeding with the withdrawal.
	* The 'withdraw' method must have a void return type.
	*
	* The 'balance' method must return the current balance.  It must have a
	* double return type.
	*
	* The 'authorize' method must set the authorization to true.  It must have a
	* void return type.
	*
	* The 'authorized' method must return the authorization.  It must have a
	* boolean return type.
	*
	*/
}
```

Here is one sequence of steps we can take to build this code:

1) declare the instance variables:

```
package code;

public class SavingsAccount {

	/* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
	*/
	private double balance;
	private boolean authorized;

	/* ... */
}
```

2) write the constructor to initialize the instance variables to sensible values:

```
package code;

public class SavingsAccount {

	/* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
	*/
	private double balance;
	private boolean authorized;

	/* The constructor of the class must initialize the balance to zero, and the
	* withdrawal authorization to false.
	*/
	public SavingsAccount() {
		balance = 0.0;
		authorized = false;
	}

	/* ... */

}
```

3) write the accessor (getter) methods balance() and authorized():

```
package code;

public class SavingsAccount {

	/* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
```

```
	*/
	private double balance;
	private boolean authorized;

	/* The constructor of the class must initialize the balance to zero, and the
	* withdrawal authorization to false.
	*/
	public SavingsAccount() {
		balance = 0.0;
		authorized = false;
	}

	/* The 'balance' method must return the current balance.  It must have a
	* double return type.
	*/
	public double balance() { return balance; }

	/* The 'authorized' method must return the authorization.  It must have a
	* boolean return type.
	*/
	public boolean authorized() { return authorized; }

	/* ... */
}
```

4) write the mutator (setter) method authorize():

```
package code;

public class SavingsAccount {

	/* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
	*/
	private double balance;
	private boolean authorized;

	/* The constructor of the class must initialize the balance to zero, and the
	* withdrawal authorization to false.
	*/
	public SavingsAccount() {
		balance = 0.0;
		authorized = false;
	}

	/* The 'balance' method must return the current balance.  It must have a
	* double return type.
	*/
	public double balance() { return balance; }

	/* The 'authorized' method must return the authorization.  It must have a
	* boolean return type.
	*/
```

```
	public boolean authorized() { return authorized; }

	/* The 'authorize' method must set the authorization to true.  It must have a
	* void return type.
	*/
	public void authorize() { authorized = true;}

	/* ... */


}
```

5) write the deposit() method:

```
package code;

public class SavingsAccount {

	/* The account object must have two private instance variables, one of type
	* double representing the current balance, and the other of type boolean
	* indicating whether the account has authorization to perform a withdrawal.
	*/
	private double balance;
	private boolean authorized;

	/* The constructor of the class must initialize the balance to zero, and the
	* withdrawal authorization to false.
	*/
	public SavingsAccount() {
		balance = 0.0;
		authorized = false;
	}

	/* The 'balance' method must return the current balance.  It must have a
	* double return type.
	*/
	public double balance() { return balance; }

	/* The 'authorized' method must return the authorization.  It must have a
	* boolean return type.
	*/
	public boolean authorized() { return authorized; }

	/* The 'authorize' method must set the authorization to true.  It must have a
	* void return type.
	*/
	public void authorize() { authorized = true;}

	/* The 'deposit' method must verify that the amount to be deposited is
	* non-negative before proceeding with the deposit.  The 'deposit' method
	* must have a void return type.
	*/
	public void deposit(double amount) {
		if (amount>=0) {
```

```
                balance = balance + amount;
            }
        }

        /* ... */
}
```

6) write the withdraw() method so it incorporates the various constraints:

```
package code;

public class SavingsAccount {

        /* The account object must have two private instance variables, one of type
        * double representing the current balance, and the other of type boolean
        * indicating whether the account has authorization to perform a withdrawal.
        */
        private double balance;
        private boolean authorized;

        /* The constructor of the class must initialize the balance to zero, and the
        * withdrawal authorization to false.
        */
        public SavingsAccount() {
            balance = 0.0;
            authorized = false;
        }

        /* The 'balance' method must return the current balance.  It must have a
        * double return type.
        */
        public double balance() { return balance; }

        /* The 'authorized' method must return the authorization.  It must have a
        * boolean return type.
        */
        public boolean authorized() { return authorized; }

        /* The 'authorize' method must set the authorization to true.  It must have a
        * void return type.
        */
        public void authorize() { authorized = true;}

        /* The 'deposit' method must verify that the amount to be deposited is
        * non-negative before proceeding with the deposit.  The 'deposit' method
        * must have a void return type.
        */
        public void deposit(double amount) {
            if (amount>=0) {
                balance = balance + amount;
            }
        }

        /* The 'withdraw' method must verify that the amount to be withdrawn is
```

```
	* non-negative, that the amount is no greater than the current balance, and
	* that the withdrawal is authorized before proceeding with the withdrawal.
	* The 'withdraw' method must have a void return type.
	*/
	public void withdraw(double amount) {
		if (amount>=0 && amount<=balance && authorized) {
			balance = balance - amount;
			authorized = false;
		}
	}

}
```