Design an ATM

This project should take approximately two hours to complete.

Write a program that allows a user to deposit or get money from an ATM. Once the program is started, it should accept input until the END command is sent. Input from the user should be in the form of text commands, as specified in the **Usage** section of each command.

The program should:

1. Store the amount of money that is in the machine. The machine initially contains $10,000.
2. Store the ATM user's current balance and transaction history.
    a. The ATM user should have an initial balance of $500.
    b. The ATM user should have no preexisting transaction history.
3. Handle any invalid requests with relevant error handling and messaging to the user. After error messaging, allow the program to continue as normal.
4. Receive and dispense money.
5. Write unit tests for the withdraw, deposit, balance, and transaction history functionality.

Specification

WITHDRAW

Removes money from the ATM user's balance. The machine only contains $20 bills, so the withdrawal amount must be a multiple of 20.

Usage: `withdraw <value>`

If the ATM user has enough money for the withdrawal, the program should print the balance after withdrawal in the format:

```
Amount dispensed: $<x>
Current balance: <balance>
```

If the ATM user has some money but not enough for the withdrawal, the full amount requested should still be dispensed. However, an additional $5 should be removed from their balance, and the following should be printed:

```
Amount dispensed: $<x>
You have been charged an overdraft fee of $5. Current balance: <balance>
```

This $5 fee must show up in the transaction history as a separate transaction that occurs immediately after the withdrawal transaction.

The ATM can't dispense more money than it contains. If in the above two scenarios the machine contains less money than was requested, the withdrawal amount should be adjusted to be the amount in the machine and this should be prepended to the printed message:

```
Unable to dispense full amount requested at this time.
```

If instead there is no money in the machine, the printed message should be this and only this:

```
Unable to process your withdrawal at this time.
```

If the ATM user's balance is $0 or less than $0, do not perform any checks against the available money in the machine, do not process the withdrawal, and print only this:

```
Your account is overdrawn! You may not make withdrawals at this time.
```

DEPOSIT

Adds money to the ATM user's balance. The deposited amount does not need to be a multiple of 20, nor does it have to be a whole dollar amount; for example, `deposit 10.36` is valid. Depositing money does *not* add money to the ATM itself.

Usage: `deposit <value>`

Prints the ATM user's balance after the deposit is made in the format:

```
Current balance: <balance>
```

BALANCE

Prints the ATM user's current balance.

Usage: `balance`

Prints the ATM user's balance in the format:

```
Current balance: <balance>
```

HISTORY

Prints the ATM user's transaction history.

Usage: `history`

If there is no history, prints:

```
No history found
```

Otherwise, prints the transaction history in reverse chronological order (most recent transaction first) in the format:

```
<date> <time> <amount> <balance after transaction>
```

For example:

```
2020-02-04 13:04:22 -20.00 140.67
2020-02-04 13:04:01 60.44 160.67
2020-02-04 13:03:49 35.00 100.23
```

END

Shuts down the server.

Usage: `end`

Prints nothing, and ends the program.