CSC 242: Introduction to Computer Science II

Lab exercise set 5

Due Wednesday, April 29th, 11:59 pm

Logistics

In order to receive full credit for the lab, you must attend the session, remain in the lab for at least 45 minutes, and submit a file that contains solutions to all these exercises.

You are encouraged to work in groups on lab exercises. If you do work with someone, you must include the name(s) of your collaborator(s) at the top of the file you submit. For more information about collaboration policies in this class, see the Academic Integrity policy.

If you complete the lab exercise early, please review Chapter 9 in the textbook and work on the fourth assignment or study for the midterm exam. You must remain in the lab for at least 45 minutes to earn full credit. You are only allowed to work on assignments with at most two-other people, either directly or indirectly, who must be formally identified as part of your assignment submission. Please see the assignment description and the course Academic Integrity pledge for the full description of the process you must follow when completing assignments. If you need additional help on the assignment, please ask the teaching assistant.

Submitting the exercises

You must submit your solutions to the exercises using the lab 5 dropbox on the D2L site. Submit only a single Python file (e.g. **csc242lab5.py**) with each of the completed functions and classes for the lab exercises in it. Submissions after the deadline listed above will be automatically rejected by the system. See the syllabus for the grading policy.

Grading

The lab session is worth 10 points. If you complete the lab exercises before the end of the lab session, please work on the fourth assignment. Remember that the rules for collaboration on assignments is different from labs. Please review the Academic Integrity policy for more information. If you have questions about the assignment, please ask the teaching assistant for help.

Problem 1:

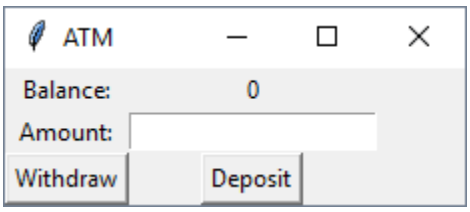
Develop a class BankAccount that supports these methods:

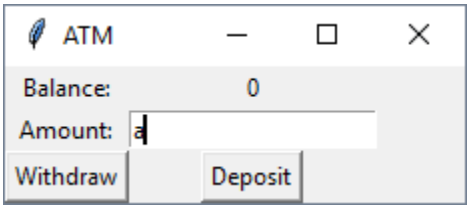
- __init__():Initializes the bank account balance to the value of the input argument, or to 0 if no input argument is given
- withdraw(): Takes an amount as input and withdraws it from the balance. Balance cannot go below 0. Returns true or false if successful.
- deposit(): Takes an amount as input and adds it to the balance. Returns true or false if successful
- balance(): Returns the balance on the account

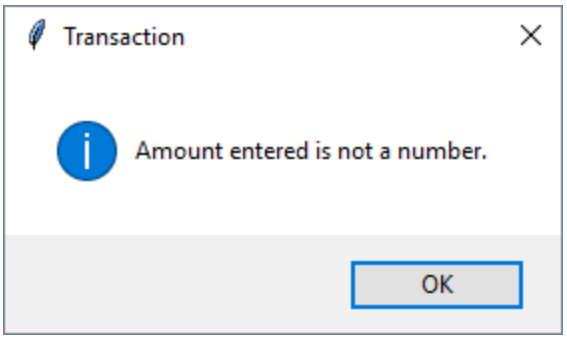
```
>>> b=BankAccount()
>>> b.withdraw(100)
False
>>> b.balance()
0
>>> b.deposit(1000)
True
>>> b.balance()
1000
>>>
```

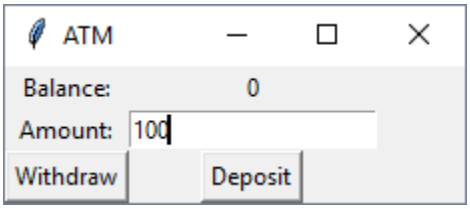
Problem 2:

Take the class from problem 1 and use it in a GUI class named ATM that has the following behaviors. You must use the class from Problem 1 to manipulate the account. The Entry box should clear after each transaction.

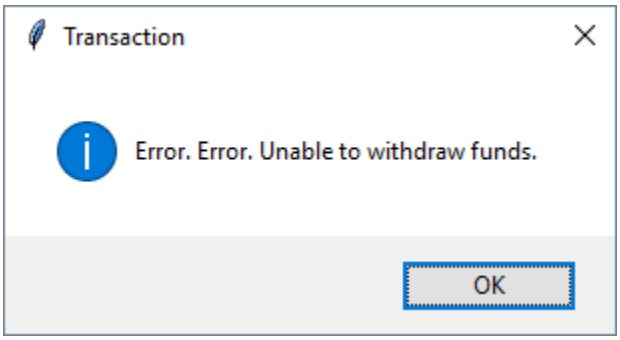


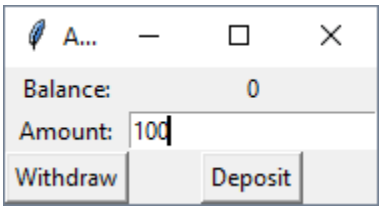




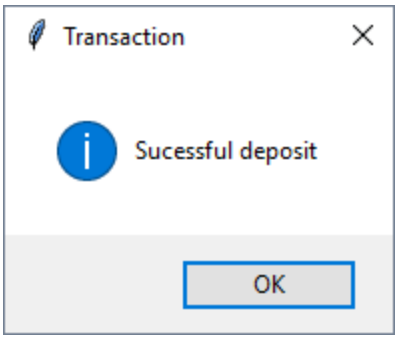


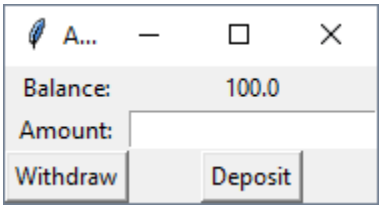
Click withdraw:





Click Deposit:


Transaction
Sucessful deposit
OK


A...
Balance:
100.0
Amount:
Withdraw
Deposit