CSE 460
Software Analysis and Design
(Fall 2019)

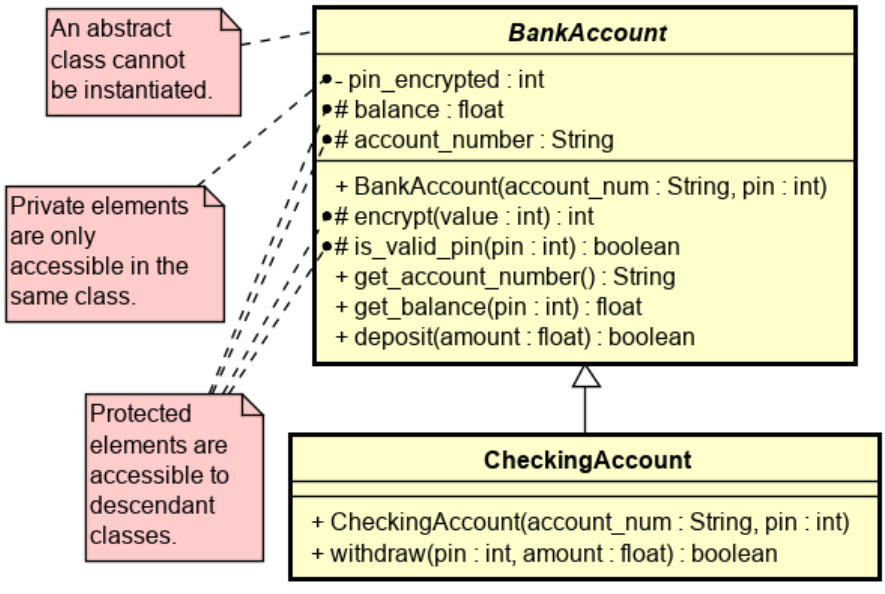
Homework #3
Assigned Date: Oct. 2, 2019
Due Date: Oct. 11, 2019

Posting ID |__|__|__|__|-|__|__|__|

Hardcopy is due at the start of the class; softcopy is due in Canvas at the start of the class.

Note 1: Your submission must include the header shown in the above rectangle. Do not put your name on your submission.
Note 2: Homework is to be done individually. You may discuss the homework problems with your fellow students, but you are NOT allowed to copy – either in part or in whole – anyone else's answers. You are also encouraged to meet the TAs and the instructor.
Note 3: All submitted materials must be legible. Text-based answers must be typed. Figures/diagrams must follow the given instructions.
Note 4: Please check the Canvas discussion board for further instructions, questions, answers, and hints.
Note 5: Convention for file name and type. Hw#-PostingID.zip (e.g., Hw3-9876-210.zip).

A software system is being developed for a bank. The bank has different types of accounts, one of which is a checking account that allows the account holder to withdraw money. An account is not allowed to have a negative balance at any time.



Each account has a specific PIN for access to the account. The PIN is not stored directly. Instead, an encrypted version of the PIN is stored for each account. Separate methods handle the encryption and validation of the PIN. IMPORTANT: Assume that the encryption algorithm will be changed during development. Only the `encrypt()` method has logic related to how to encrypt a PIN.

(a) [10 points] Specify at least one suitable invariant for the `BankAccount` class.

(b) [30 points] State useful pre-conditions for the `deposit()`, `withdraw()`, and `get_balance()` operations. Use logical Boolean expressions to define the pre and post-conditions. Hint: Think about ways the methods might be used incorrectly.

(c) [10 points] State at least one post-condition for the `withdraw()` method.

(d) [50 points] Implement the `BankAccount` and the `CheckingAccount` classes using the Java programming language.

NOTE: The content of the course slides for the "Design-by-Contract" (see Ch. 3: Classes and Objects entry appearing under Lecture Materials) is based on the "Applying 'design by contract'.pdf" article. This was noted in the class held on Oct. 2nd.

Submission: Use Hw#-PostingID.zip for the name of a zip file that contains the softcopy of the report, source code, and readme files. Turn in the paper copy for the answers to parts (a), (b), and (c) in class.

Create Java implementation for the classes. Your code must be compatible with Java 8, Standard Edition. Please follow these guidelines exactly.

- The program must implement the classes shown in the diagram above, with the same class, attribute, and method names. Your program will be graded by another program, so spelling and punctuation are important.
- For the `encrypt()` method, use the formula $output = (input \cdot 4) - 7$. This is a placeholder formula which should not appear outside of the `encrypt()` method.
- All accounts have a balance of 0 when created.
- Use the pre-conditions above to place restrictions for `get_balance()`, `deposit()`, and `withdraw()`. If an attempt is made to make a deposit that is not allowed, print the message "Deposit cancelled." to stdout. If a withdrawal cannot be completed, print the message "Withdrawal cancelled." If a request to view the balance must be cancelled, print the message "The balance could not be viewed." If an operation cannot be completed, no change is made to the account balance. Include the period in the message, and end the message with a new line. These operations return `True` if successful; otherwise `False` is returned.
- If your program has any "package" statements, remove them all before turning in the program. All your code should be in the same Java package.
- You do not need to print your program (part d)) on paper. You do need to turn in a paper copy of the report for parts (a), (b) and (c). Turn in both the report and the program to Blackboard.
- Please, do not copy these instructions to your report. -2 points for printing the Submission instruction.

Hints for this assignment:

- JUnit is helpful for testing. Git is helpful for versioning. Neither is required.
- Ask the professor and the TA your questions. We love answering questions before the assignment is due, instead of afterward. See our office hours on Canvas, and check the discussion forum.