Task 2.1 (UML Use Cases) (2 Points)
Design an information system for a bank company. Consider the following use cases:

- A potential customer wants to open a bank account.
- A customer uses an ATM to withdraw money from his/her account.
- A customer applies for a credit.
- A customer makes a money transfer using the web site of the bank.
- A customer closes an account.

Sketch a use case diagram that covers these use cases (compare slides 60/62 of chapter 2). Define sub-use cases where appropriate and describe their relationship (‘uses’ or ‘extends’).

In addition, give a detailed textual description of the second use case (customer uses ATM) including a detailed description of the procedure and error situations (e.g. wrong PIN number entered, not enough money in ATM; compare slide 61 of chapter 2).

Task 2.2 (UML Structure Diagram) (2 Points)
Sketch a structure diagram for the scenario from exercise 2.1.
Consider the following object types:

- Accounts are either private accounts or business accounts. A private account is used by one or more persons. A person may have multiple accounts. A Business account is used by a company. Persons and companies have addresses.
- A bank company has branches and ATMs. Branches and ATMs have addresses. An ATM is operated by a branch.
- A bank company has employees. Employees are persons and work in a specific branch.
- Transactions are money transfers, cash deposits, or cash withdrawals. Money transfers can be made via the Internet or in written form at a branch of the bank. Cash withdrawals can be made at an ATM or at a branch of the bank. If they are made at a branch, an employee is responsible.

Develop a UML structure diagram that models the above domain and give a few reasonable attributes and operation signatures for the classes.

---

1Automated teller machine, German: Geldautomat
2German: Überweisungen
3Bareinzahlungen
4Barabhebungen
Task 2.3 (ER-Modelling)

Each of the following ER diagrams contains (at least) one mistake. Identify what went wrong and correct the error.

a) The ER diagram below is intended to model the following scenario:

A course may or may not use a textbook. By definition, a textbook is a book that is used in some course. A course may not use more than five books. Instructors teach from two to four courses. A course is taught by exactly one instructor. Instructors base their lectures (adopt) on the content of all textbooks, that are used for their courses.

For simplicity we left out the entities’ attributes (do not consider this as an error!).

b) The diagram below is intended to model part of an international student and university database.

Task 2.4 (Ternary Relationships)

Consider the following ER diagram as presented in the last tutorial session.

It models a university database that deals with courses (like e.g. ‘Introduction to Databases’, ‘Data Communication’, etc.) held by instructors in certain semesters. Please note, that courses of the same title held in different semesters are modelled as one entity in this approach.

Mike wants to use a software tool to design this ER model. Unfortunately, it does not support ternary relationships. He regards them as superfluous anyway and proposes the following diagram:

a) Show that Mike is not able to model the offer of courses by instructors in certain semester as precisely as in the ternary relationship case. Illustrate this by giving a suitable database state\(^5\) for the first model and the corresponding database state for Mike’s model. Now start with this second state and derive a corresponding state according to the first model. Choose an initial database state such that this round trip does not reach its starting state.

b) What is the minimal size (in the number of instances in the ternary relationship) of such an counterexample? Prove your answer\(^6\).

c) What assumption about the domain has to be made (reflected by a change in the marked cardinality constraint) to make Mike’s simplification valid?

---

\(^5\)A database state contains the instances of all entity types together with the relationship type instances (tuples of the keys of the participating entities).

\(^6\)Recommendation: Make a constructive proof.