ACIT4420 Assignment 1 Report

The assignment uses a simple banking system imitation using Python classes and inheritance. Based on the given problem description, we are to achieve three classes: BankAccount as a base, SavingsAccount that inherits from BankAccount and adds interest, and CheckingAccount that inherits from Bank Account and overrides withdraw to intake a withdraw fee.

Question 1: How inheritance was used to avoid redundancy?

From our base class which is BankAccount, we have deposit, withdraw, and account_info within BankAccount. This is the key in avoiding redundancy, while also using super() function. Our subclasses do not have to reimplement them and only goes back to our BankAccount and add or change it. This is the idea from classes "blueprint + reuse." For example, SavingsAccount only adds interest_rate and apply_interest to update and no deposit or withdraw.

Question 2: How the withdraw() method was overridden in CheckingAccount?

So if we take a look at BankAccount's withdraw, the base rule first checks if the balance is enough. It only subtracts the amount. But CheckingAccount replaces the parent's withdraw including the transaction fee. The new rule for withdraw costs a fixed fee needs to be overridden because the base rule is not enough. This is a demonstration of Lecture 4's Method Overriding so Python searches order through subclass then base class.

Question 3: Brief explanation of any optimizations or design decisions.

To avoid weird float rounding, we uses *decimal* so the result can be exact. Every deposit and withdraw also adds a checks that the amount must be positive or it detects value errors. Another design is adding "Insufficient funds" to avoid automatic python errors.