**14.6.3 Άσκηση: Εφαρμογή Προηγμένων Εννοιών Κλάσεων**

**Σενάριο – Άσκηση (class\_BankAccount.py):**

Έχετε προσληφθεί για να δημιουργήσετε ένα απλό σύστημα τραπεζικού λογαριασμού. Το σύστημα πρέπει να υποστηρίζει δύο τύπους λογαριασμών: `***SavingsAccount***` και `***CheckingAccount***`. Και οι δύο τύποι λογαριασμών πρέπει να έχουν κοινές λειτουργίες όπως `***deposit()***`, `***withdraw()`*** και `***get\_balance()***`, αλλά κάθε τύπος λογαριασμού έχει κάποια μοναδικά χαρακτηριστικά.

1. Ορίστε μια βασική κλάση με το όνομα `***BankAccount***` με τις ακόλουθες μεθόδους:

- ***`\_\_init\_\_(self, account\_number, initial\_balance)***`: Αρχικοποιεί τον λογαριασμό με έναν αριθμό λογαριασμού και έναν αρχικό υπόλοιπο.

- `***deposit(self, amount)***`: Προσθέτει το καθορισμένο ποσό στο υπόλοιπο του λογαριασμού.

- `***withdraw(self, amount)*`:** Αφαιρεί το καθορισμένο ποσό από το υπόλοιπο του λογαριασμού.

- `***get\_balance(self)***`: Επιστρέφει το τρέχον υπόλοιπο του λογαριασμού.

2. Δημιουργήστε μια υποκλάση με το όνομα `***SavingsAccount***` που κληρονομεί από την `***BankAccount***`. Προσθέστε τα ακόλουθα χαρακτηριστικά:

- Μια μεταβλητή κλάσης με το όνομα `***interest\_rate***` που είναι ίση με 0.05 (5%).

- Αντικαταστήστε τη μέθοδο `***get\_balance()***` για να υπολογίζει και να επιστρέφει το υπόλοιπο συν το επιτόκιο που έχει αποκτηθεί βάσει του επιτοκίου.

3. Δημιουργήστε μια άλλη υποκλάση με το όνομα `***CheckingAccount***` που κληρονομεί από την `***BankAccount***`. Προσθέστε τα ακόλουθα χαρακτηριστικά:

- Μια μεταβλητή κλάσης με το όνομα `***transaction\_fee***` που είναι ίση με 1.0.

- Αντικαταστήστε τη μέθοδο `***withdraw()***` για να αφαιρεί το τέλος συναλλαγής από το υπόλοιπο του λογαριασμού.

4. Πραγματοποιήστε διάφορες λειτουργίες όπως καταθέσεις, αναλήψεις και ερωτήσεις υπολοίπου και παρατηρήστε τα αποτελέσματα.

**Λύση:**

***class BankAccount:***

***def \_\_init\_\_(self, account\_number, initial\_balance):***

***self.account\_number = account\_number***

***self.balance = initial\_balance***

***def deposit(self, amount):***

***self.balance += amount***

***def withdraw(self, amount):***

***self.balance -= amount***

***def get\_balance(self):***

***return self.balance***

***class SavingsAccount(BankAccount):***

***interest\_rate = 0.05***

***def get\_balance(self):***

***interest = self.balance \* self.interest\_rate***

***return self.balance + interest***

***class CheckingAccount(BankAccount):***

***transaction\_fee = 1.0***

***def withdraw(self, amount):***

***self.balance -= amount + self.transaction\_fee***

***# Δοκιμή του κώδικα***

***savings = SavingsAccount("SAV001", 1000)***

***savings.deposit(500)***

***savings.withdraw(200)***

***print("Υπόλοιπο Λογαριασμού Αποταμίευσης:", savings.get\_balance())***

***checking = CheckingAccount("CHK001", 2000)***

***checking.deposit(100)***

***checking.withdraw(50)***

***print("Υπόλοιπο Λογαριασμού Έλεγχου:", checking.get\_balance())***

***Έξοδος:***

***Υπόλοιπο Λογαριασμού Αποταμίευσης: 1352.5***

***Υπόλοιπο Λογαριασμού Ελέγχου: 2049.0***

Σε αυτήν τη λύση, καθορίζουμε την κλάση `**BankAccount**` ως τη βασική κλάση με κοινές μεθόδους όπως `**deposit**()`, `**withdraw**()` και `**get\_balance()**`. Οι υποκλάσεις `**SavingsAccount**` και `**CheckingAccount**` κληρονομούν από την `**BankAccount**` και προσθέτουν τα δικά τους μοναδικά χαρακτηριστικά.

Η υποκλάση `**SavingsAccount**` υπολογίζει και επιστρέφει το υπόλοιπο συν τα επιτόκια που έχουν συσσωρευτεί βάσει του επιτοκίου. Η υποκλάση `CheckingAccount` αφαιρεί ένα τέλος συναλλαγής από το υπόλοιπο του λογαριασμού όταν γίνεται μια ανάληψη.

Στη συνέχεια, δημιουργούμε παραδείγματα τόσο για το `**SavingsAccount**` όσο και για το `**CheckingAccount**` και πραγματοποιούμε διάφορες λειτουργίες για να δοκιμάσουμε τον κώδικα. Η έξοδος εμφανίζει τα υπόλοιπα που προκύπτουν μετά την εκτέλεση των λειτουργιών.

Μπορείτε ελεύθερα να τροποποιήσετε τον κώδικα και να δοκιμάσετε διάφορα σενάρια για να εξερευνήσετε περαιτέρω τα έννοια των μεταβλητών κλάσης, της ενθυλάκωσης και της πολυμορφίας.