Python Assignment: Object-Oriented Bank Account System

# 🎯 Objective

Design and implement a class-based system to represent and manage bank accounts using Python. This project will help you practice using:  
- Classes and objects  
- Class and instance methods  
- Properties and encapsulation  
- Static and class-level attributes  
- Working with time and datetime  
- Input validation  
- Custom object formatting

# 💡 Task Overview

You will create a Python program with at least one main class: Account. Optionally, you may also use a helper class such as TimeZone.  
  
Your program should support:  
- Creating a bank account with account details  
- Making deposits and withdrawals  
- Adding monthly interest  
- Generating and parsing transaction confirmation codes

# 🧪 Project Structure by Grade Level

## ✅ Grade 3 – Basic Functionality

- Account class with:  
 - \_\_init\_\_ method accepting account number, first name, last name, and optional balance  
 - Properties for first name, last name (read/write)  
 - Property for full name (computed)  
 - Read-only balance property  
- Deposit and withdrawal methods  
 - Withdrawals cannot exceed balance  
 - Return confirmation code (basic string, like D-acc\_number-ID)  
- A class-level interest rate (hardcoded or default)  
- Simple method to add interest (without confirmation code)

## ✅ Grade 4 – Intermediate Functionality

- Everything from Grade 3, plus:  
- Enforce non-negative balance (cannot directly set balance)  
- Confirmation code format: TRANSACTIONTYPE-ACCOUNTNUMBER-UTC-TIMESTAMP-ID  
- Include an incrementing transaction ID (class level counter)  
- Implement pay\_interest method that calculates and deposits interest  
- Time of transaction stored and shown in confirmation code (UTC)

## ✅ Grade 5 – Advanced Functionality

- Everything from Grade 4, plus:  
- Create and use a TimeZone class (name + UTC offset in hours)  
- Preferred time zone passed when creating an account  
- Store transactions with actual datetime values  
- Implement parse\_confirmation\_code(code, preferred\_timezone\_offset):  
 - Returns structured object with attributes: transaction\_code, account\_number, time\_utc, time (converted), transaction\_id  
- Use UTC everywhere for storing, display in preferred timezone  
- Bonus: Allow the interest rate to be changed using a class method  
- Use float for balances (mention that Decimal is better in real-world situations)

# 🧹 Optional Simplifications / Adjustments

- Remove TimeZone class and use integer offset  
- Skip timezone conversion and use local time only  
- Use dict or namedtuple for parsed confirmation code instead of custom object

# 🧪 Test & Deliverables

- Include a small test script showing usage of all features  
- Clearly comment your code  
- (Optional but encouraged): Use unittest to test deposit/withdraw and interest

# ✨ Prep Phrasing (Instructions to Students)

In this assignment, you will design a Python class to represent a bank account system. You will implement functionality such as deposits, withdrawals, interest calculation, and tracking transactions through confirmation numbers.  
  
Your code should follow object-oriented principles and include data encapsulation, class-level attributes, and proper method usage. You are encouraged to be creative, follow clean coding practices, and test your implementation.  
  
Depending on the level of complexity you reach, you will receive a grade of 3, 4, or 5. See the project brief for details on what each level requires.