**A6: Inheritance & Polymorphism**

**Purpose**

Review and reinforcement of **inheritance, polymorphism, static, const,** classes, separate compilation, makefile, and pointer-related topics.

**Description**

In this assignment, you will utilize the Account class that we created in the class and define two child classes named **Savings** and **Checking**, respectively and apply polymorphism by meeting the following requirements.

R1: Account class

class BankAccount

{

private:

double\* balance;

double\* interestRate;

static int count;

public:

BankAccount();

BankAccount(double, double);

virtual ~BankAccount(); //must, delete the memory in the heap

double get\_balance() const;

double get\_interest\_rate() const;

void set\_balance(double);

void set\_interest\_rate(double);

void print(ostream&);

void deposit(double);

void withdraw(double);

static int get\_count();

virtual void update() = 0; //pure virtual function

};

Note that your program stores the values of balance and interest\_rate in the heap and the destructor is a virtual function

R2: Checking class

Implement the update() by applying the annual interest rate of 1.5% on the balance over $1000

R3: Savings class

Implement the update() by applying the annual interest rate of 3% on *any* balance.

R4. Test program

In your main(), you will demonstrate how polymorphism in C++ is implemented and tested. You need to find a way to implement the requirements below using polymorphism.

1. Using a loop or loops, create three savings and three checking accounts using the following balances in the heap.

|  |  |
| --- | --- |
| Account | Initial Balance |
| Savings#1 | $500.50 |
| Savings#2 | $700.70 |
| Savings#3 | $900.90 |
| Checking#1 | $200.10 |
| Checking#2 | $400.30 |
| Checking#3 | $600.60 |

1. In each year for five years and for every bank account, apply a deposit of $300 three times and a withdrawal of $150.50 as long as withdrawal is valid. If the balance is 0 and withdrawal occurs, displays "Invalid withdrawal" for the sake of error handling, though this will not happen in our program. Also, apply the annual interest rate *at the end of each year*.
2. At the end of the fifth year, display the balance of each account.



**Submission requirements: Same as before with the following emphasis.**

* **In the self-scoring rubric, write both earned points and reasons if you deduct any points**
* **A5\_Lastname\_Firstname.zip that include all the source code, makefile, and the rubric.**