Challenge 2: Implement an Account Class Using Pure Virtual Functions

In this challenge, we'll implement an account class along with two derived classes saving and current.

WE'LL COVER THE FOLLOWING

Problem Statement

Write a code that has:

- A **parent** *class* named `Account`.
  - Inside it *define*:
    - a protected float member `balance`
  - We have three pure virtual functions:
    - `void Withdraw(float amount)`
    - `void Deposit(float amount)`
    - `void printBalance()`
- Then, there are **two derived** *classes*
  - `Savings` class
    - has a *private* member `interest_rate` set to 0.8
    - `Withdraw(float amount)` deducts *amount* from *balance* with *interest rate*

        - `Deposit(float amount)` adds *amount* in *balance* with *interest_rate*
        - `printBalance()` displays the balance in the *account*
    - `Current` class
        - `Withdraw(float amount)` deducts *amount* from *balance*
            - `Deposit(float amount)` adds *amount* in *balance*
            - `printBalance()` displays the balance in the *account*`

Input

- In `Savings` class, `balance` is set to **50000** in parametrized constructor of `Savings` object called by `Account` class
- In `Current` class, `balance` is set to **50000** in parametrized constructor of `Current` object called by `Account` class

Here's a sample result which you should get.

Sample Input

```
Account * acc[2];
acc[0] = new Savings(50000);
acc[0]->Deposit(1000);
acc[0]->printBalance();

acc[0]->Withdraw(3000);
acc[0]->printBalance();

acc[1] = new Current(50000);
acc[1]->Deposit(1000);
acc[1]->printBalance();

acc[1]->Withdraw(3000);
acc[1]->printBalance();
```

Sample Output

```
Balance in your saving account: 51800
Balance in your saving account: 46400

Balance in your current account: 51000
Balance in your current account: 48000
```

Coding Exercise

Implement the code in the **problem** tab.

Good Luck!

Exercise | Solution

```
#include <iostream>
using namespace std;

// Write classes code here
// make base class functions virtual

int main() {
  // make instances of classes here
  // call their traits functions here
  return 0;
}
```

Show Hint

Solution Review

- We have implemented `Account` class which has **balance** float variable, and three pure virtual functions **Deposit(float amount), Withdraw(amount)** and **printBalance()**

- Now implement `Savings` and `Current` classes inherited publicly from

`Account` class

- `Savings` has private float **interest_rate** variable and functions:
  - `Withdraw(float amount)` deducts *amount* from *balance* with *interest_rate*
  - `Deposit(float amount)` adds *amount* in *balance* with *interest_rate*
  - `printBalance()` displays the balance in the *account*
- `Current` has functions:
  - `Withdraw(float amount)` deducts *amount* from *balance*
  - `Deposit(float amount)` adds *amount* in *balance*
  - `printBalance()` displays the balance in the *account*`
- Create *Savings* and *Current* object by calling parametrized constructors of the classes and print their balance by calling their respective functions

---

In the next chapter, we'll learn about the advanced concepts of **Composition, Aggregation and Association.**