Object-Oriented Programming with C++
Polymorphism - Part 2

ENSIA 2024-2025

Objective

The objective of the Coin Toss Game is to accumulate a balance of exactly **100.0 DA** by tossing different coins.

Game Rules

- The game uses three types of coins: `coin10`, `coin20`, and `coin50`.
- Each coin has a specific value:
  - `coin10` is worth 10 DA.
  - `coin20` is worth 20 DA.
  - `coin50` is worth 50 DA.
- At the beginning of the game, your balance is set to 0.
- During each round, all three coins are tossed simultaneously.
- If a coin lands heads-up, its value is added to your balance. If it lands tails-up, no change occurs.
- The game continues until your balance reaches or exceeds 100.0 DA.
- If your balance exactly equals 100.0 DA, you win the game.
- If your balance exceeds 100.0 DA, you lose the game.

Gameplay

- The game starts with a balance of 0.0 DA.
- The program simulates the tossing of each coin (`coin10`, `coin20`, and `coin50`) in every round.
- After each coin toss, the result is displayed: whether the coin landed heads-up or tails-up.
- If a coin lands heads-up, its value is added to the balance, and the updated balance is displayed.
- If the balance reaches or exceeds 100.0 DA, the game ends:
  - If the balance equals 100.0 DA, a message will be displayed indicating that you have **won** the game.
  - If the balance exceeds 100.0 DA, a message will indicate that you have **lost** the game.

Note: The program uses dynamic casting and polymorphism to handle the different types of coins and their behaviors during the coin toss.

Enjoy playing the Coin Toss Game and try to accumulate exactly **100.0 DA** to win!

Abstract Class for Coins

You can use the following abstract class to define derived classes for each specific coin.

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>

class Coin {
public:
    virtual bool toss() const = 0; // heads or tails
    virtual double getValue() const = 0;
};
```

You are required to implement three derived classes: `Coin10`, `Coin20`, and `Coin50`, each overriding the `toss()` and `getValue()` methods appropriately.

Coin Evaluation Function

Use the following function to simulate the toss of a coin and evaluate its outcome. This function is standalone and will be invoked by the `main()` function.

```
void evaluateToss(Coin* coin) {
    // This function should:
    // 1. Simulate the toss using coin->toss()
    // 2. Print whether it landed heads or tails
    // 3. If heads, add coin->getValue() to the balance
}
```

Main Function

In your `main()` function, you should:

- Instantiate one object for each coin type.
- Initialize the balance to 0.0.
- Loop until the balance reaches or exceeds 100.0 DA.
- Call `evaluateToss()` for each coin in each round.
- Display the balance and outcome after each round.