Reflection log

Class Definition:

The `MySavings` class is a simple console-based application that simulates a piggy bank where users can add coins (pennies, nickels, dimes, quarters) or withdraw money from the bank. It provides a user-friendly menu for interacting with the `PiggyBank` class, which keeps track of the total amount in the bank. The program continues to run until the user decides to exit by typing `Q` or `q`.

Main Method:

The main method serves as the entry point of the program, handling the user interface and interaction. It continuously displays a menu with options to:

- Show the total amount in the bank.
- Add specific coins (penny, nickel, dime, quarter).
- Withdraw money from the bank.
- Exit the program.

The program uses a `switch` statement to handle user input and calls methods from the `PiggyBank` class to perform the operations. A `while` loop ensures that the program keeps running until the user opts to quit by entering `Q`.

```
public static void main(String[] args) {

    Scanner scanner = new Scanner(System.in);

    PiggyBank piggyBank = new PiggyBank();

    boolean running = true;


    while (running) {

        System.out.println("1. Show total in bank.");

        System.out.println("2. Add a penny.");

        System.out.println("3. Add a nickel.");
```

```
        System.out.println("4. Add a dime.");
        System.out.println("5. Add a quarter.");
        System.out.println("6. Take money out of bank.");
        System.out.println("Enter Q to quit.");
        System.out.print("Enter your choice: ");

        String choice = scanner.nextLine().trim();

        switch (choice) {
            case "1":
                System.out.printf("Total in bank: $%.2f%n",
piggyBank.getTotal());
                break;
            case "2":
                piggyBank.addPenny();
                System.out.println("Added a penny.");
                break;
            case "3":
                piggyBank.addNickel();
                System.out.println("Added a nickel.");
                break;
            case "4":
                piggyBank.addDime();
```

```
                System.out.println("Added a dime.");
                break;
            case "5":
                piggyBank.addQuarter();
                System.out.println("Added a quarter.");
                break;
            case "6":
                piggyBank.takeMoneyOut();
                break;
            case "Q":
            case "q":
                running = false;
                System.out.println("Exiting the application.
Goodbye!");
                break;
            default:
                System.out.println("Invalid choice. Please try
again.");
                break;
        }
    }

    scanner.close();
```

```
}
```

Constructor:

In this code, the constructor is not explicitly defined within the `MySavings` class, but the `PiggyBank` object (`piggyBank`) is instantiated at the start of the `main` method. This object handles the logic for adding and removing money from the bank.

```
PiggyBank piggyBank = new PiggyBank();
```

This constructor call creates a new `PiggyBank` object that is used throughout the program to interact with the bank's balance.

PiggyBank Class:

The `PiggyBank` class, although not included in the provided code, is assumed to contain methods like:

- `getTotal()`: Returns the total amount in the bank.
- `addPenny()`, `addNickel()`, `addDime()`, `addQuarter()`: Methods to add the respective coins to the total.
- `takeMoneyOut()`: Allows money to be withdrawn from the bank.

These methods are called from the `MySavings` class to perform various operations.

User Interaction and Logic:

The program relies on a simple text menu system. Users are prompted to choose an option:

- **Show total in bank:** Displays the current balance.
- **Add a coin:** Users can add a penny, nickel, dime, or quarter.
- **Take money out of the bank:** This option likely prompts the user to specify an amount to withdraw (though the actual implementation is missing from the provided code).
- **Quit:** The user can exit the program.

The use of a `switch` statement makes it easy to handle different choices and ensures the program responds appropriately to user input. If the user enters an invalid choice, they are prompted to try again.

Challenges Encountered:

1. **Missing Withdrawal Logic:**
   The program allows the user to choose to withdraw money from the bank (option "6"), but the actual functionality for withdrawing money is not provided in the `PiggyBank` class (based on the code provided). Implementing this feature could involve asking the user for an amount and deducting it from the balance.
2. **Error Handling:**
   While the program handles invalid menu choices by prompting the user to try again, there is no error handling in place for other potential issues, such as negative or invalid withdrawal amounts (if the withdrawal functionality were implemented). Adding checks for such cases would improve robustness.