EXERCISE VIIII

Go as far as you can!

9.1 What is the output of the following program?

```
public class Foo {
  private static int i = 0;
  private static int j = 0;

  public static void main(String[] args) {
    int i = 2;
    int k = 3;

    {
      int j = 3;
      System.out.println("i + j is " + i + j);
    }

    k = i + j;
    System.out.println("k is " + k);
    System.out.println("j is " + j);
  }
}
```

9.2 Describe the role of the **this** keyword. What is wrong in the following code?

```
public class C {
  private int p;

  public C() {
    System.out.println("C's no-arg constructor invoked");
    this(0);
  }

  public C(int p) {
    p = p;
  }

  public void setP(int p) {
    p = p;
  }
}
```

9.3 (*The* `MyInteger` *class*) Design a class named `MyInteger`. The class contains:

- An `int` data field named `value` that stores the `int` value represented by this object.
- A constructor that creates a `MyInteger` object for the specified `int` value.
- A `get` method that returns the `int` value.
- Methods `isEven()`, `isOdd()`, and `isPrime()` that return `true` if the value is even, odd, or prime, respectively.
- Static methods `isEven(int)`, `isOdd(int)`, and `isPrime(int)` that return `true` if the specified value is even, odd, or prime, respectively.
- Static methods `isEven(MyInteger)`, `isOdd(MyInteger)`, and `isPrime(MyInteger)` that return `true` if the specified value is even, odd, or prime, respectively.
- Methods `equals(int)` and `equals(MyInteger)` that return `true` if the value in the object is equal to the specified value.
- A static method `parseInt(char[])` that converts an array of numeric characters to an `int` value.
- A static method `parseInt(String)` that converts a string into an `int` value.

Draw the UML diagram for the class. Implement the class. Write a client program that tests all methods in the class.

9.4 (*Game: ATM machine*) Use the **Account** class created in Exercise 8.7 to simulate an ATM machine. Create ten accounts in an array with id **0**, **1**, . . . , **9**, and initial balance $100. The system prompts the user to enter an id. If the id is entered incorrectly, ask the user to enter a correct id. Once an id is accepted, the main menu is displayed as shown in the sample run. You can enter a choice **1** for viewing the current balance, **2** for withdrawing money, **3** for depositing money, and **4** for exiting the main menu. Once you exit, the system will prompt for an id again. So, once the system starts, it will not stop.

```
Enter an id: 4 ↵Enter

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 ↵Enter
The balance is 100.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 2 ↵Enter
Enter an amount to withdraw: 3 ↵Enter
```

```
Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 ↵Enter
The balance is 97.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 3 ↵Enter
Enter an amount to deposit: 10 ↵Enter

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 ↵Enter
The balance is 107.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 4 ↵Enter

Enter an id:
```