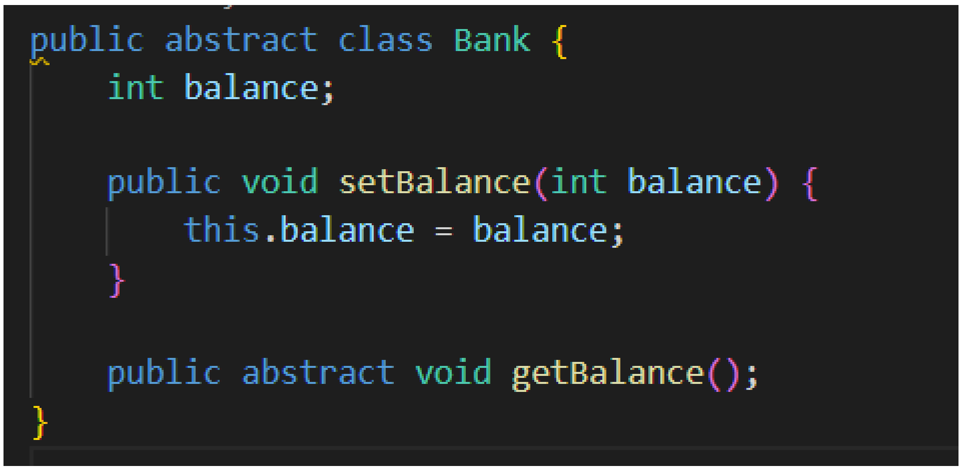
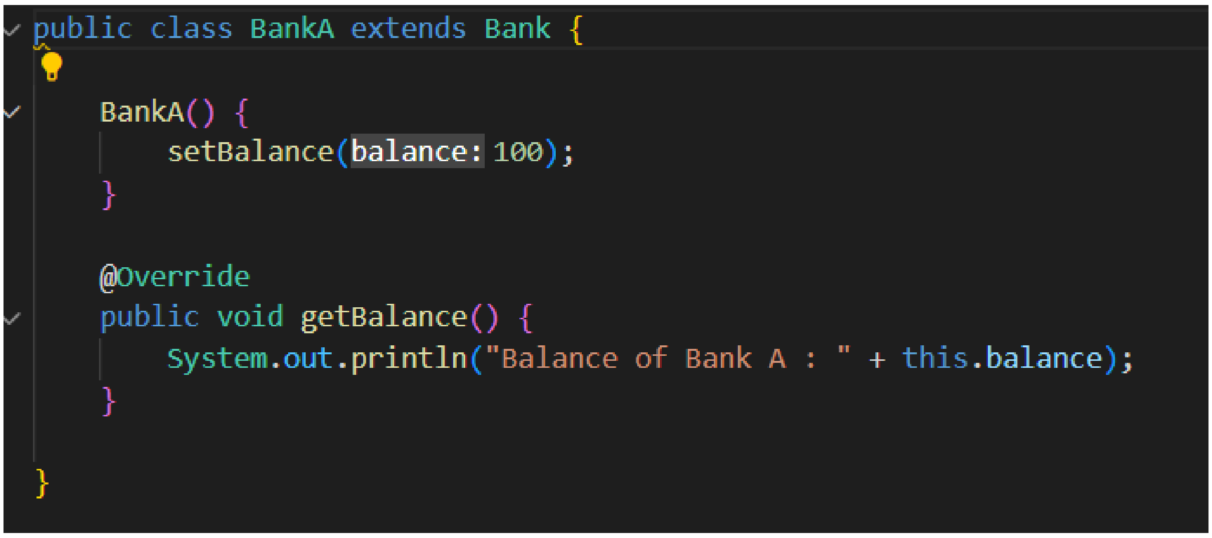
1) Bank Class



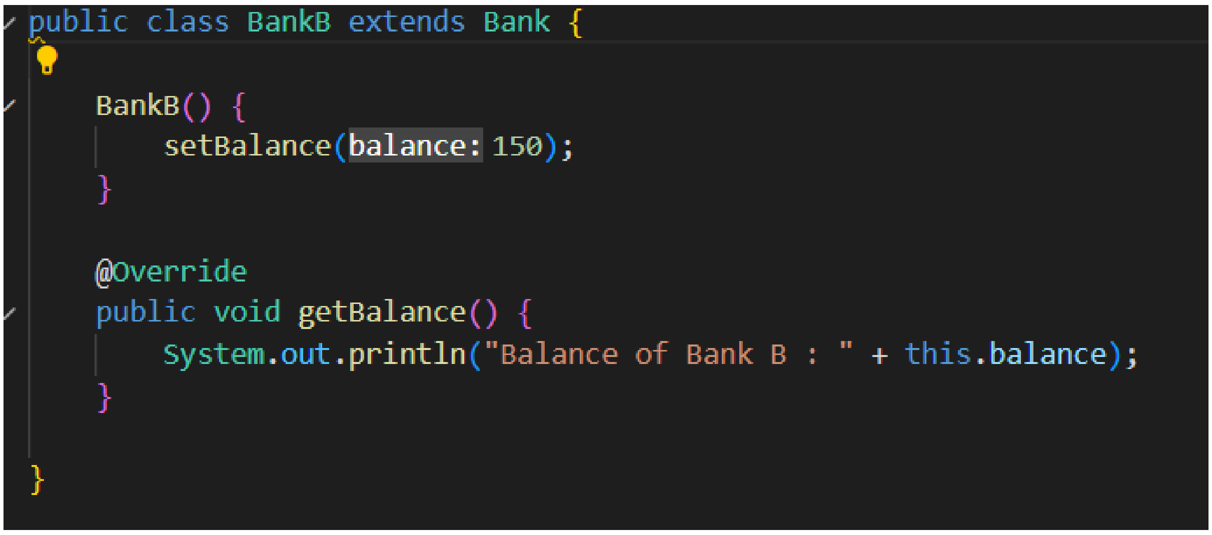
```java
public abstract class Bank {
    int balance;

    public void setBalance(int balance) {
        this.balance = balance;
    }

    public abstract void getBalance();
}
```

BankA Class



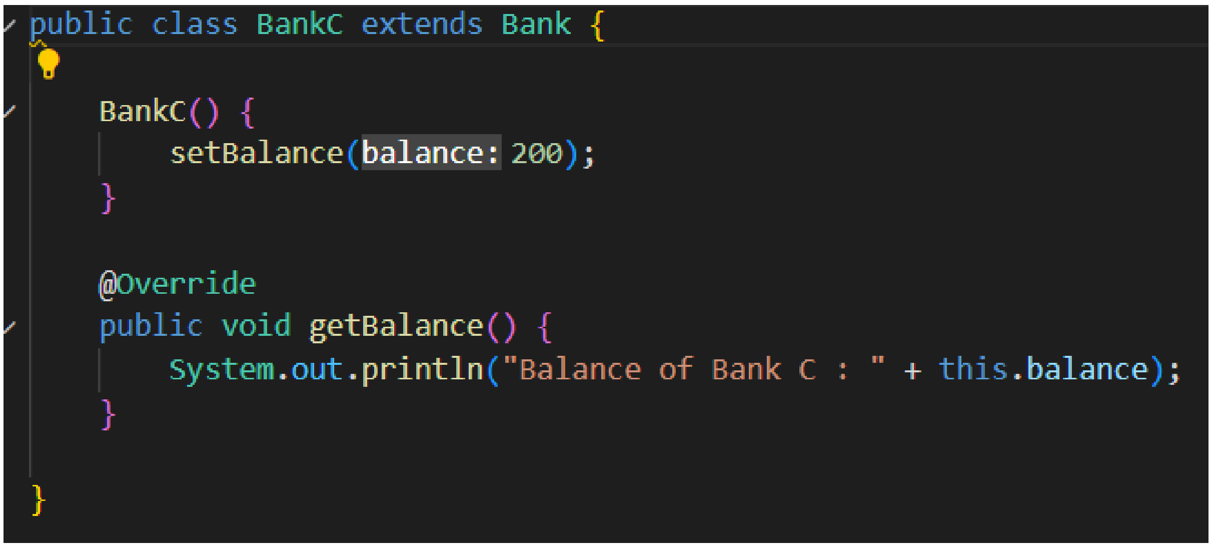
```java
public class BankA extends Bank {

    BankA() {
        setBalance(balance: 100);
    }

    @Override
    public void getBalance() {
        System.out.println("Balance of Bank A : " + this.balance);
    }

}
```

BankB Class



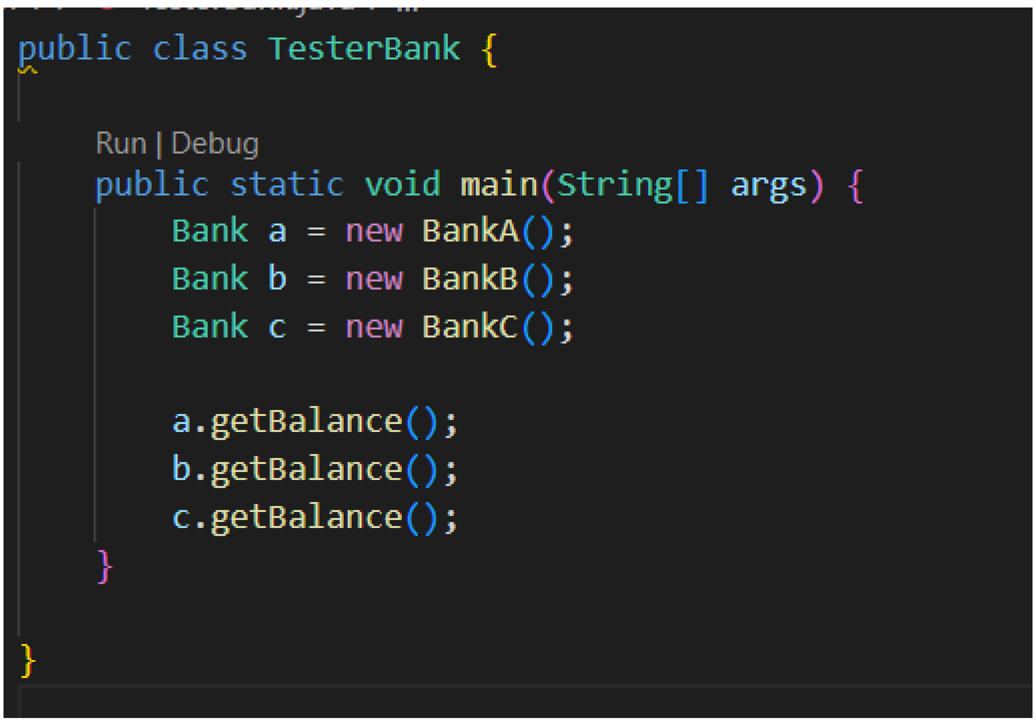
```java
public class BankB extends Bank {

    BankB() {
        setBalance(balance: 150);
    }

    @Override
    public void getBalance() {
        System.out.println("Balance of Bank B : " + this.balance);
    }

}
```

BankC Class



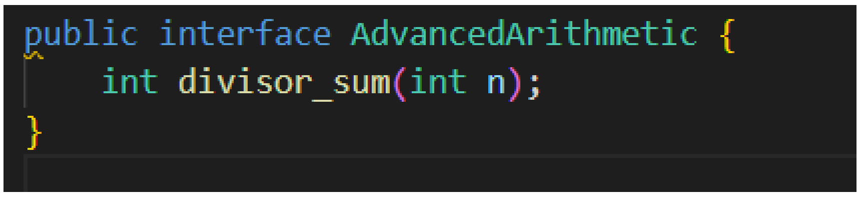
```java
public class BankC extends Bank {

    BankC() {
        setBalance(balance: 200);
    }

    @Override
    public void getBalance() {
        System.out.println("Balance of Bank C : " + this.balance);
    }

}
```

TesterBank Class



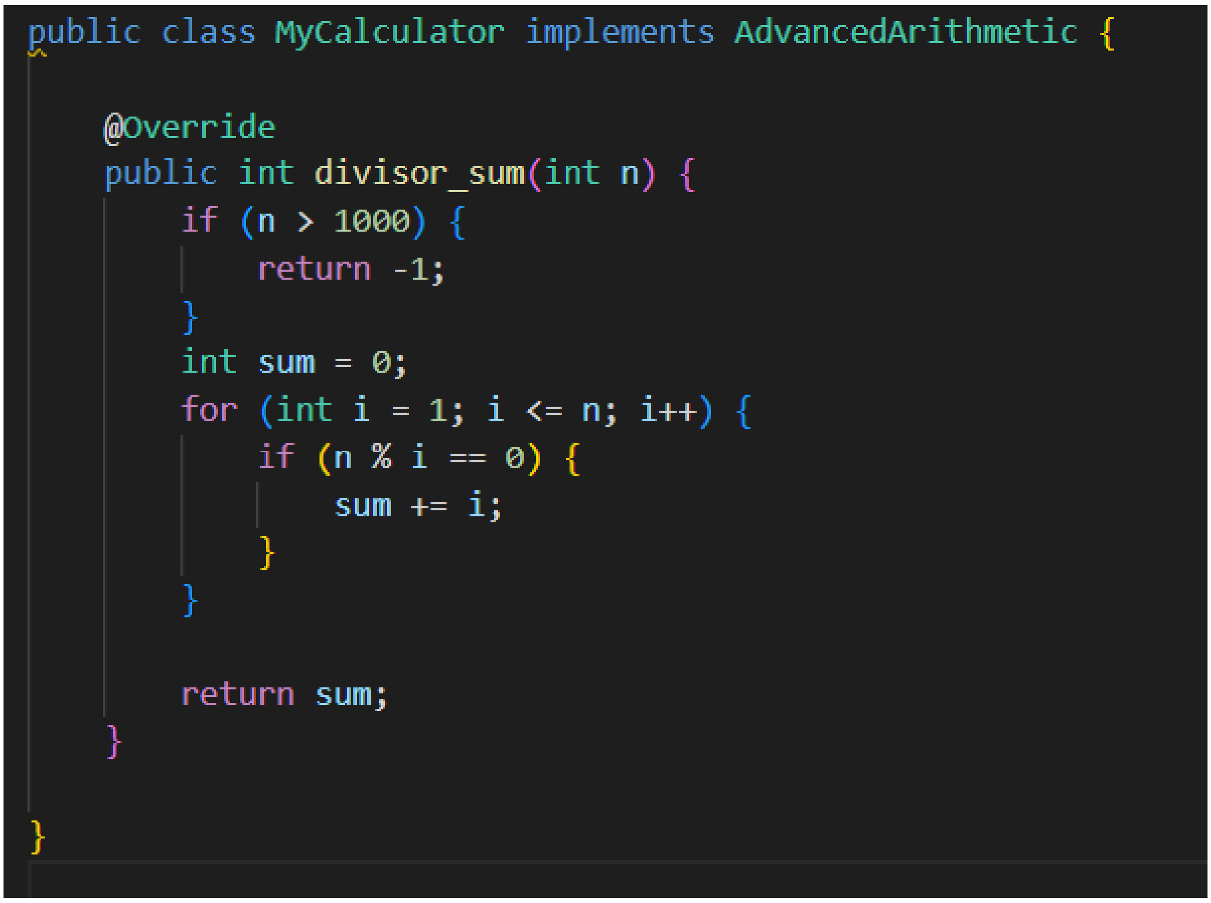
```java
public class TesterBank {

    Run | Debug
    public static void main(String[] args) {
        Bank a = new BankA();
        Bank b = new BankB();
        Bank c = new BankC();

        a.getBalance();
        b.getBalance();
        c.getBalance();
    }

}
```

2) AdvancedArithmetic Interface



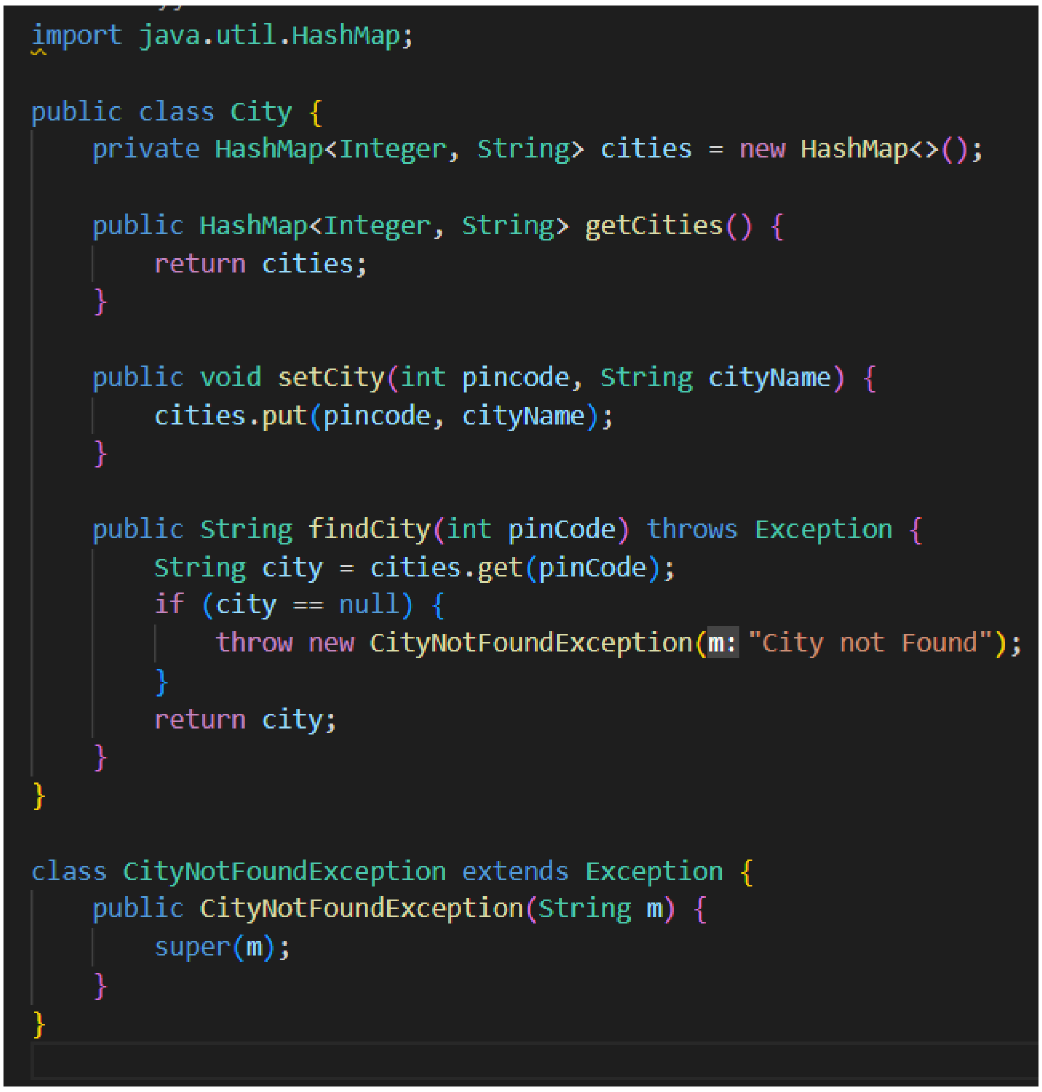
```java
public interface AdvancedArithmetic {
    int divisor_sum(int n);
}
```

MyCalculator class



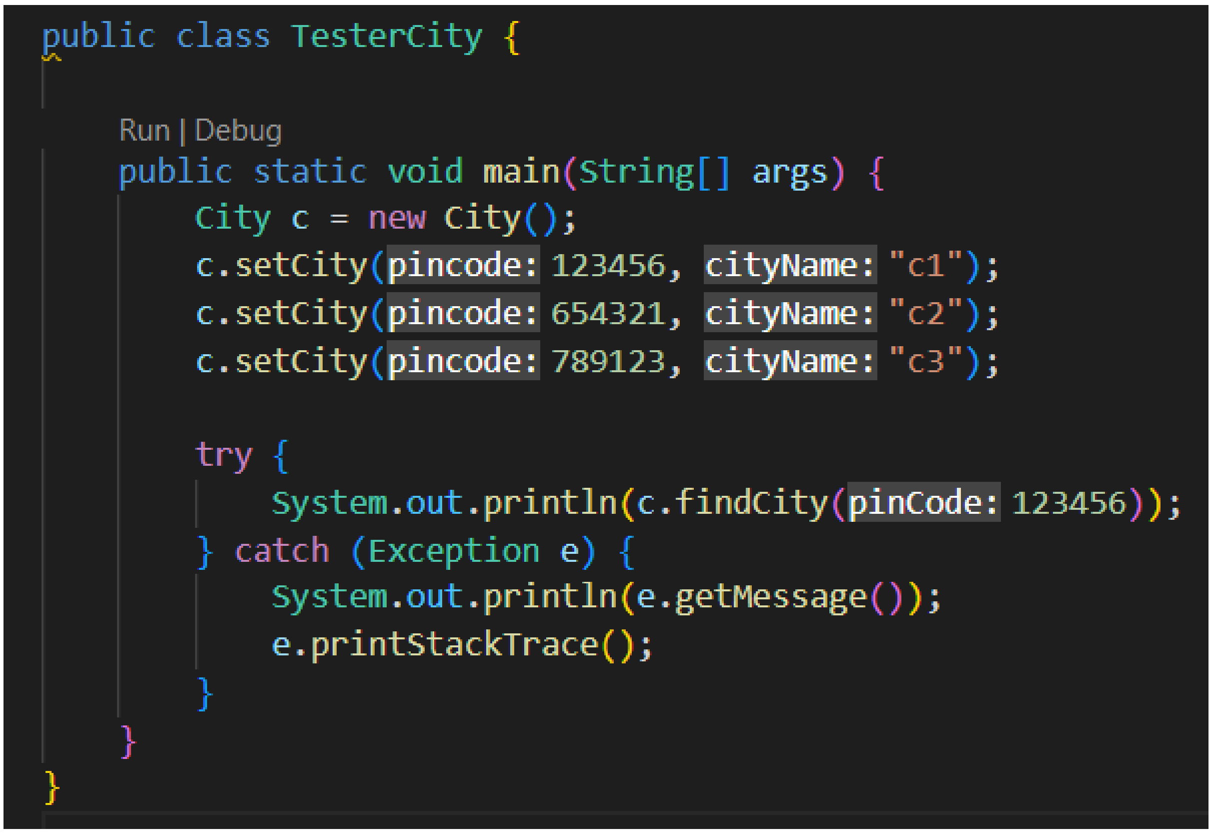
```java
public class MyCalculator implements AdvancedArithmetic {

    @Override
    public int divisor_sum(int n) {
        if (n > 1000) {
            return -1;
        }
        int sum = 0;
        for (int i = 1; i <= n; i++) {
            if (n % i == 0) {
                sum += i;
            }
        }

        return sum;
    }

}
```

3) City Class



```java
import java.util.HashMap;

public class City {
    private HashMap<Integer, String> cities = new HashMap<>();

    public HashMap<Integer, String> getCities() {
        return cities;
    }

    public void setCity(int pincode, String cityName) {
        cities.put(pincode, cityName);
    }

    public String findCity(int pinCode) throws Exception {
        String city = cities.get(pinCode);
        if (city == null) {
            throw new CityNotFoundException(m: "City not Found");
        }
        return city;
    }
}

class CityNotFoundException extends Exception {
    public CityNotFoundException(String m) {
        super(m);
    }
}
```

TesterCity Class



```
public class TesterCity {

    Run | Debug
    public static void main(String[] args) {
        City c = new City();
        c.setCity(pincode: 123456, cityName: "c1");
        c.setCity(pincode: 654321, cityName: "c2");
        c.setCity(pincode: 789123, cityName: "c3");

        try {
            System.out.println(c.findCity(pinCode: 123456));
        } catch (Exception e) {
            System.out.println(e.getMessage());
            e.printStackTrace();
        }
    }
}
```