```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try {
            System.out.print("Enter a number: ");
            double number = scanner.nextDouble();

            if (number < 0) {
                throw new IllegalArgumentException("Error: Cannot calculate the square root of a negative number.");
            }

            double result = Math.sqrt(number);
            System.out.println("Square Root: " + result);
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        } catch (Exception e) {
            System.out.println("Error: Invalid input. Please enter a valid number.");
        } finally {
            scanner.close();
        }
    }
}
```

input

```
Enter a number: -12
Error: Cannot calculate the square root of a negative number.
```

```java
import java.util.Scanner;

class InvalidPinException extends Exception {
    public InvalidPinException(String message) {
        super(message);
    }
}

class InsufficientBalanceException extends Exception {
    public InsufficientBalanceException(String message) {
        super(message);
    }
}

public class Main{
    private static final int CORRECT_PIN = 1234;
    private static double balance = 3000;

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try {
            System.out.print("Enter PIN: ");
            int enteredPin = scanner.nextInt();

            if (enteredPin != CORRECT_PIN) {
                throw new InvalidPinException("Error: Invalid PIN.");
            }

            System.out.print("Withdraw Amount: ");
            double amount = scanner.nextDouble();

            if (amount > balance) {
```

```
        if (amount > balance) {
            throw new InsufficientBalanceException("Error: Insufficient balance.");
        }

        balance -= amount;
        System.out.println("Withdrawal Successful! Remaining Balance: " + balance);
    } catch (InvalidPinException | InsufficientBalanceException e) {
        System.out.println(e.getMessage());
    } catch (Exception e) {
        System.out.println("Error: Invalid input. Please enter a valid number.");
    } finally {
        System.out.println("Current Balance: " + balance);
        scanner.close();
    }
}
```

```
Enter PIN: 123
Error: Invalid PIN.
Current Balance: 3000.0
```

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

class CourseFullException extends Exception {
    public CourseFullException(String message) {
        super(message);
    }
}

class PrerequisiteNotMetException extends Exception {
    public PrerequisiteNotMetException(String message) {
        super(message);
    }
}
public class Main {
    private static final int COURSE_CAPACITY = 2;
    private static Map<String, Integer> courseEnrollment = new HashMap<>();
    private static Map<String, String> coursePrerequisites = new HashMap<>();
    private static Map<String, Boolean> studentCompletedCourses = new HashMap<>();

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        coursePrerequisites.put("Advanced Java", "Core Java");
        studentCompletedCourses.put("Core Java", false);

        try {
            System.out.print("Enroll in Course: ");
            String course = scanner.nextLine();
            if (coursePrerequisites.containsKey(course)) {
                String prerequisite = coursePrerequisites.get(course);

                System.out.println("Prerequisite: " + prerequisite);
```

```
        try {
            System.out.print("Enroll in Course: ");
            String course = scanner.nextLine();
            if (coursePrerequisites.containsKey(course)) {
                String prerequisite = coursePrerequisites.get(course);

                System.out.println("Prerequisite: " + prerequisite);

                if (!studentCompletedCourses.getOrDefault(prerequisite, false)) {
                    throw new PrerequisiteNotMetException("Complete " + prerequisite +
                    " before enrolling in " + course + ".");
                }
            }

            int enrolled = courseEnrollment.getOrDefault(course, 0);
            if (enrolled >= COURSE_CAPACITY) {
                throw new CourseFullException("Course is full. Cannot enroll in " + course + ".");
            }

            courseEnrollment.put(course, enrolled + 1);
            System.out.println("Enrollment Successful! Enrolled in: " + course);

        } catch (PrerequisiteNotMetException | CourseFullException e) {
            System.out.println("Error: " + e.getClass().getSimpleName() + " - " + e.getMessage());
        } catch (Exception e) {
            System.out.println("Error: Invalid input.");
        } finally {
            scanner.close();
        }
    }
}
```

```
Enroll in Course: java
Enrollment Successful! Enrolled in: java
```