UNTAR
Universitas Tarumanagara

Terakreditasi BAN PT A (Unggul)

UNTAR untuk INDONESIA

# Object-based Programming

## Week 6 – Dealing with Errors

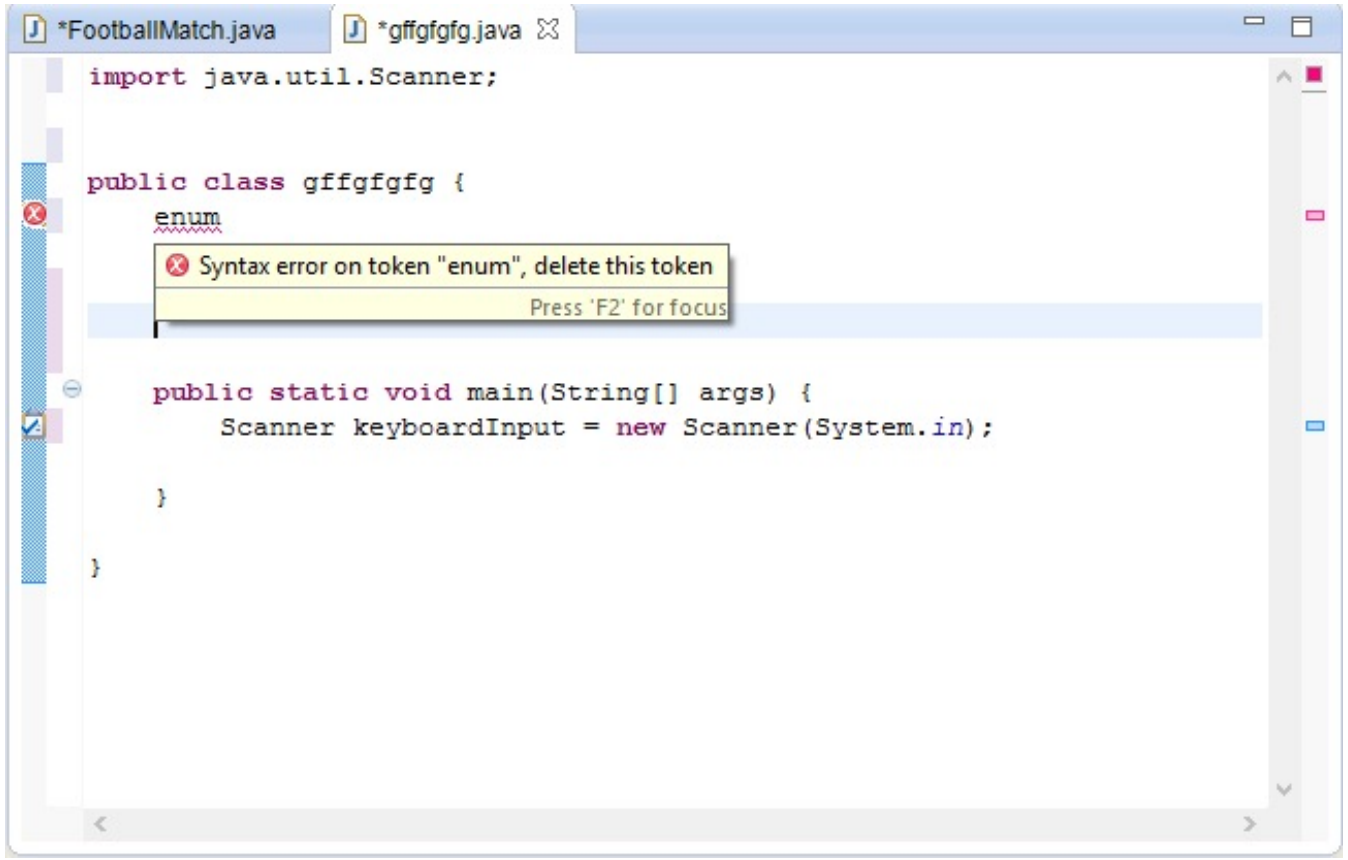
# How much effort should I put into ensuring that my software is free from errors?

# Types of errors

- Coding syntax errors
- Logic errors
- Design errors
- User errors
- Run time errors
- Environmental errors

UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Types of errors (1/6): coding syntax errors






https://stackoverflow.com/questions/35054672/syntax-error-on-token-enum-delete-this-token

# Types of errors (2/6): logic errors

```
class DateUtility {
    public boolean isLeapYear(int year) {
        return (year % 4 == 0);
    }
}
```

UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Types of errors (3/6): design errors

- The code works fine, but does not address the requirements of the user

# Types of errors (4/6): user errors

- The code works, but does not respond consistently or coherently when faced with erroneous user input

- For example, the code asks the user to "*select a menu option from 1 to 5*" and the user enters 6
  - Did the software provide a useful error mesage?

# Types of errors (5/6): run time errors

- Failure to open external assets, such as files
- Network connections
- JVM running out of memory
- etc..

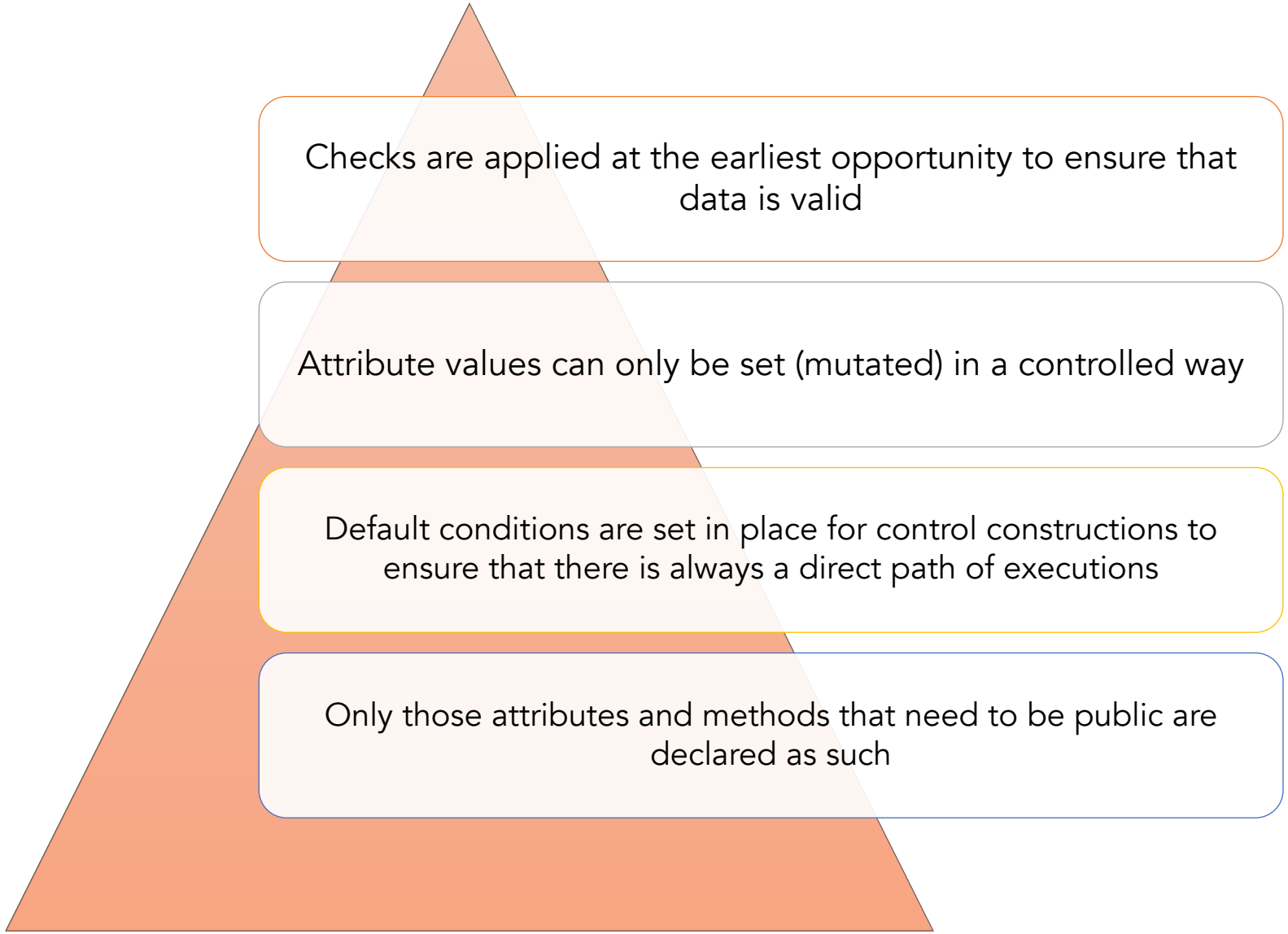
- Solution: exception handling

# Types of errors (6/6): environmental errors

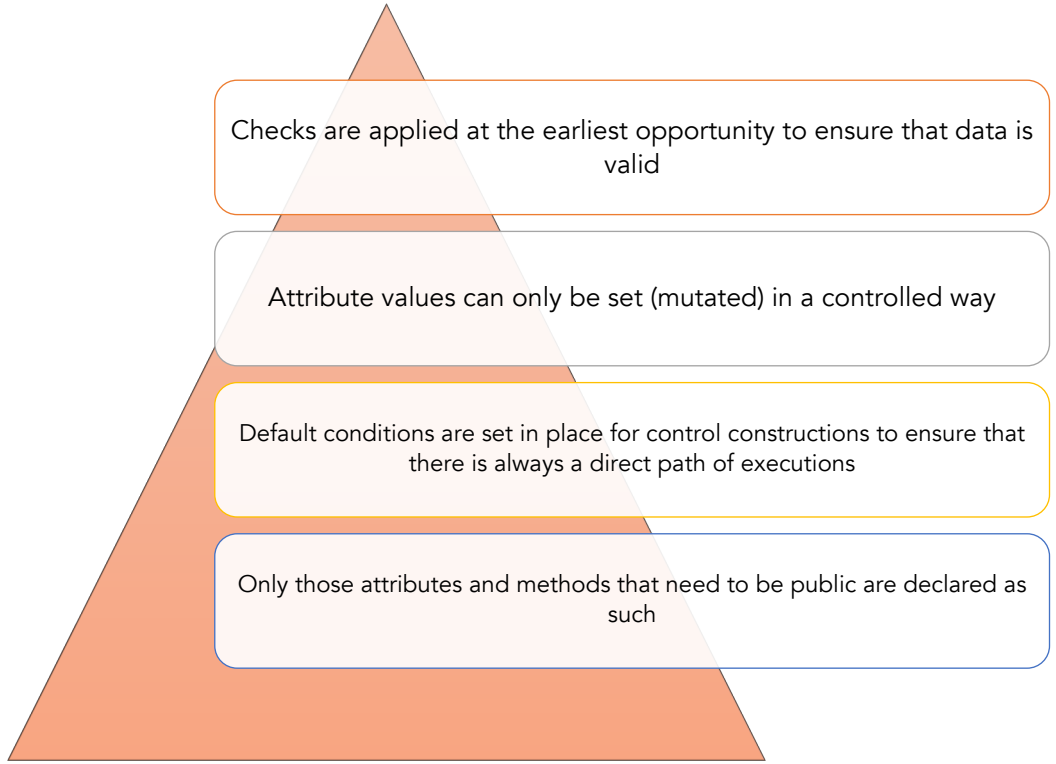
- Hardware failure
- Power outage when processing data

# Coding Defensively

prevention is better than cure


Checks are applied at the earliest opportunity to ensure that data is valid
Attribute values can only be set (mutated) in a controlled way
Default conditions are set in place for control constructions to ensure that there is always a direct path of executions
Only those attributes and methods that need to be public are declared as such

```
public class Student {
    private double ipk;

    public void setIpk(double ipk) {
        if (ipk >= 0.0 && ipk <= 4.0) {
            this.ipk = ipk;
        } else {
            System.out.println("Invalid IPK");
        }
    }
}
```



# Question #1

https://bit.ly/3B2zJrL

```
public class DefensiveCoding {
   private int x;
   private ArrayList<String> aList;

   public void poorCoding() {
      aList = new ArrayList<String>();
      aList.add(“python”);
      aList.add(“java”);
      x = 2;

      for (int y = 0; y < x; y++) {
         System.out.println(aList.get(y));
      }
   }
}
```

A

B

C

D

E

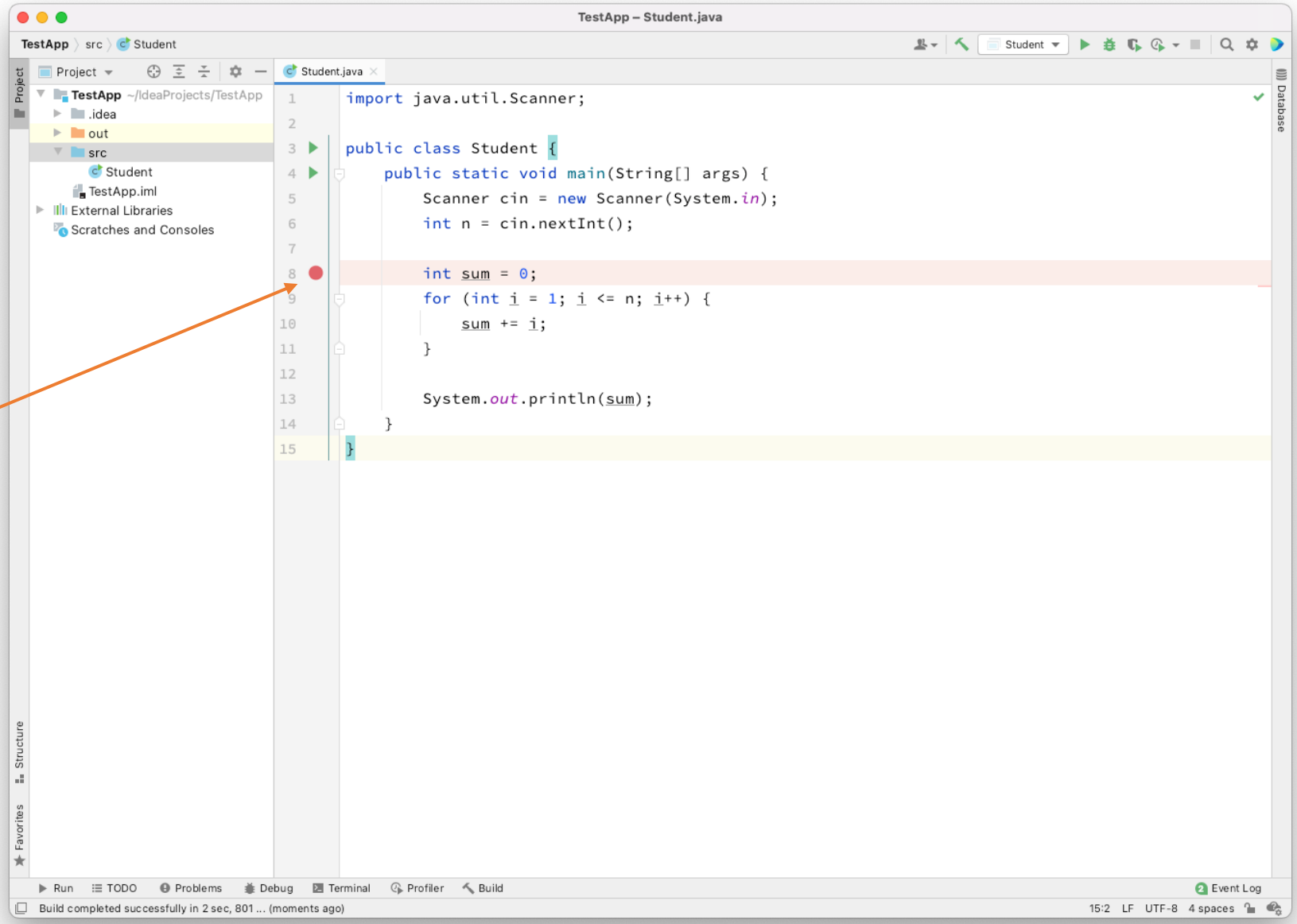
UNTAR Universitas Tarumanagara

UNTAR untuk INDONESIA

```
public class DefensiveCoding {
    private ArrayList<String> aList;

    public void poorCoding() {
        aList = new ArrayList<String>();   aList.add(“python”);    aList.add(“java”);

        // cara 1
        for (int y = 0; y < aList.size(); y++) {
            System.out.println(aList.get(y));
        }

        // cara 2
        for (String s: aList) {
            System.out.println(s);
        }
    }
}
```

# Using the Debugger Tool

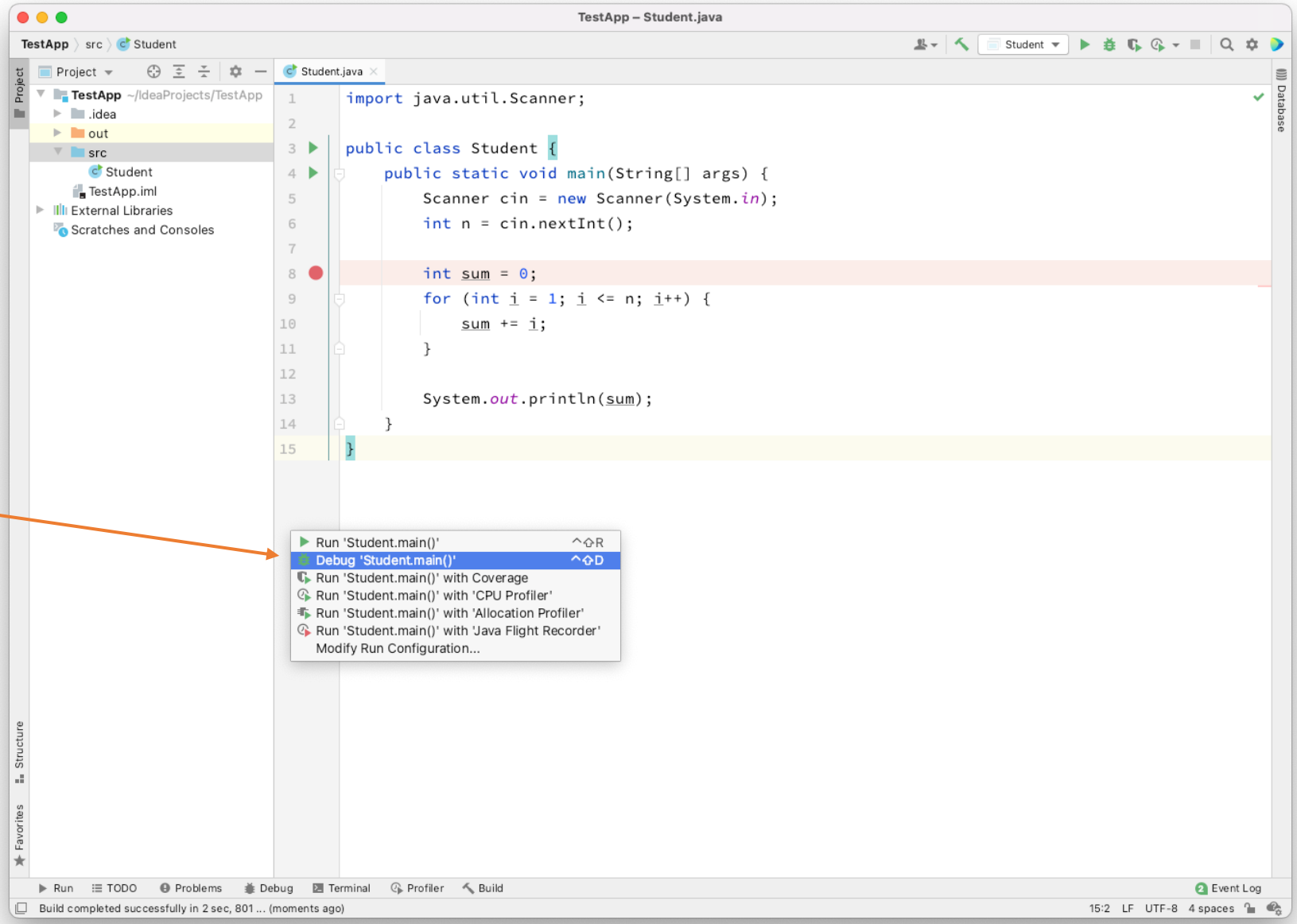
add breakpoint




UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Using the Debugger Tool

click debug

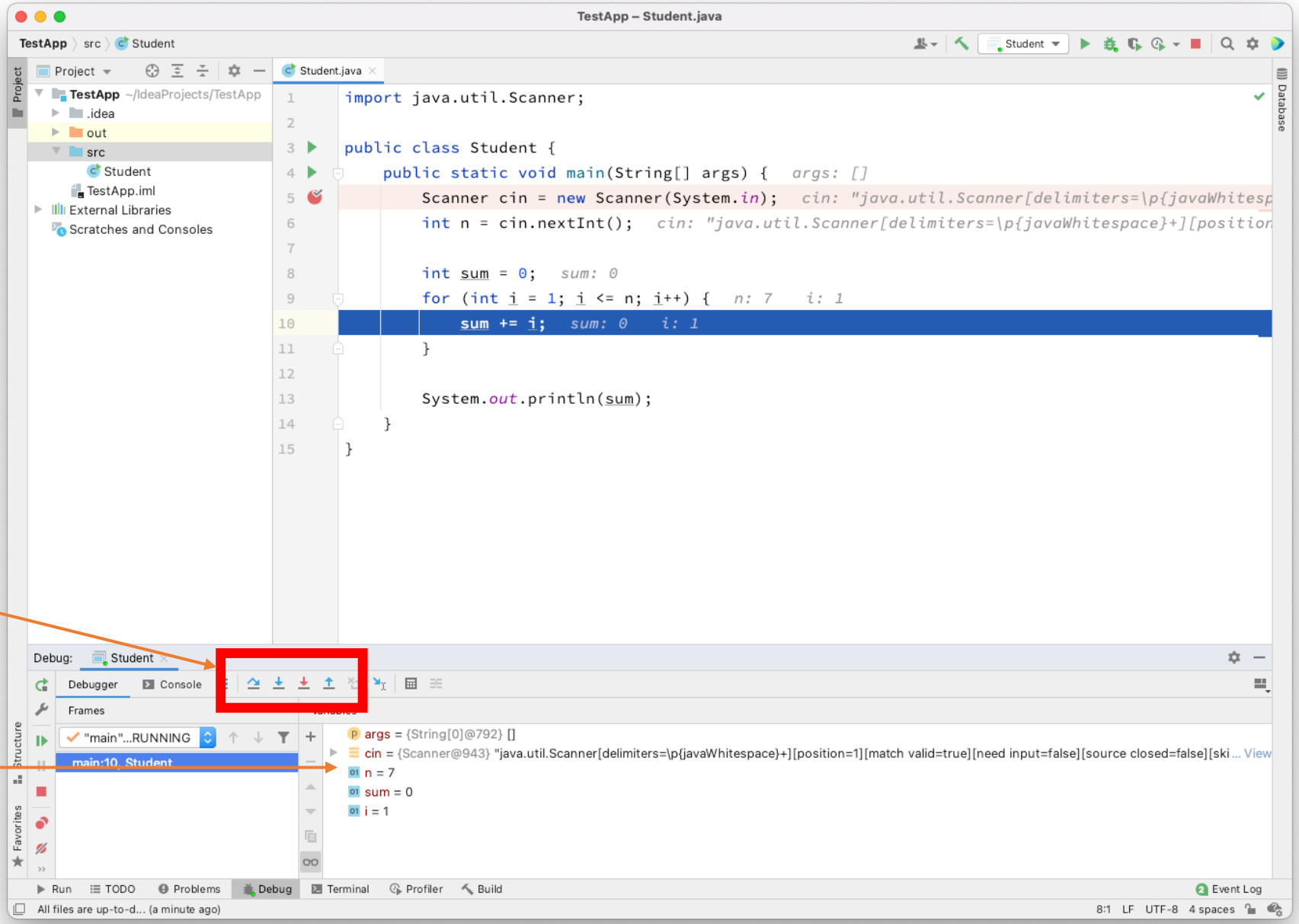



UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Using the Debugger Tool

run each steps

watch variables




UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Unit Testing

## Unit level testing

- Individual classes are tested

## System level testing

- A program as a whole is checked to see that it operates as intended

# Unit test class

- A class contained in a project that permits automated testing of one or more classes

- The test class itself does not form part of the actual functional codebase

```
public class BookReviews {
    private ArrayList<String> allReviews;
    private double rating;

    ...

    /**
     * A method to add a review to the list of reviews.
     * @param String with the text of the review
     * @param int with the rating out of 5
     * @return true if the review was accepted
     */
    public boolean addReview(String review, int myRating) {
        ...
    }
}
```

```
public class BookReviews {
   private ArrayList<String> allReviews;
   private double rating;

   ...

   /**
    * A method to add a review to the list of reviews.
    * @param String with the text of the review
    * @param int with the rating out of 5
    * @return true if the review was accepted
    */
   public boolean addReview(String review, int myRating) {
      ...
   }
}
```

The concept here is that we can enter a review and a rating only if:

- the rating is between 0 and 5 inclusive
- the review is not empty text

```
import static org.junit.Assert.*;
import org.junit.*;

public class BookReviewsTest {
    ...

    @Test
    public void addReviewReturnsTrue() {
        BookReviews b = new BookReviews(“Fishing”, “Smith”);
        assertEquals(
            b.addReview(“A splendid read! Highly recommended!”, 5),
            true
        );
    }
}
```

UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

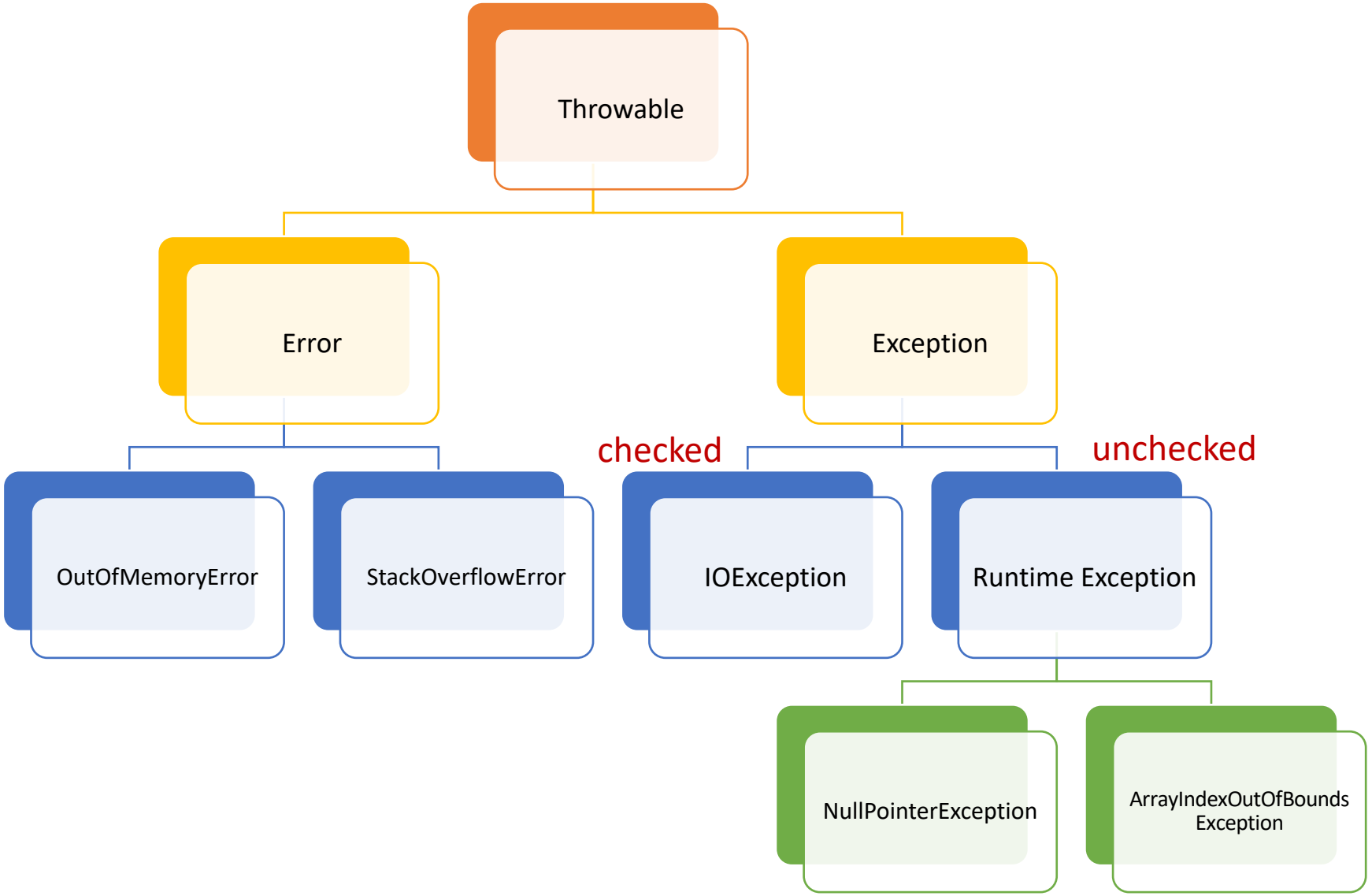
```
import static org.junit.Assert.*;
import org.junit.*;

public class BookReviewsTest {
    ...

    @Test
    public void blankReviewShouldReturnsFalse() {
        BookReviews b = new BookReviews(“Fishing”, “Smith”);
        assertEquals(
            b.addReview(“    ”, 3),
            false
        );
    }
}
```

# Exception Handling

```
class DemoException {
    public static void main(String[] args) {
        int d = 0;
        int a = 10 / d;
    }
}
```

```
class DemoException {
    public static void main(String[] args) {
        int d = 0;
        int a = 10 / d;
    }
}
```

```
java.lang.ArithmeticException: /  by zero
    at DemoException.main(DemoException.java:4)
```


Throwable
Error
Exception
checked
unchecked
OutOfMemoryError
StackOverflowError
IOException
Runtime Exception
NullPointerException
ArrayIndexOutOfBounds Exception

# Question #2

https://bit.ly/3B2zJrL

```
public class XYZ {
    public static void main(String[] args) {
        ArrayList<Integer> arr;
        arr.add(1);
        arr.add(2);
        arr.add(3);
        arr.add(4);
        arr.add(5);

        System.out.println(arr.get(14));
    }
}
```

What kind of Exception will occur if we run the code?

A. ArrayIndexOutOfBoundsException
B. IOException
C. ArithmeticException
D. SQLException
E. NullPointerException

# Checked vs Unchecked Exception

- **Unchecked exception**
  - Compiler does not enforce (check) that you handle them explicitly
  - e.g. NullPointerException, ArithmeticException

- **Checked exception**
  - You must handle the exception or the code won't compile
  - e.g. IOException, SQLException

# Handling Exception (1)

```
public void openFileAndReadLines() {
   try {
      FileReader f = new FileReader("C:\\test.txt");
      BufferedReader br = new BufferedReader(f);

      String line = br.readLine();
      while (line != null) {
         System.out.println(line);
         line = br.readLine();
      }

      br.close();  f.close();
   }
   catch (IOException e) {
      System.out.println("Error when opening the file.");
      System.out.println(e.getMessage());
   }
}
```

UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA

# Handling Exception (2)

```
public void openFileAndReadLines() throws IOException {
    FileReader f = new FileReader("C:\\test.txt");
    BufferedReader br = new BufferedReader(f);

    String line = br.readLine();
    while (line != null) {
        System.out.println(line);
        line = br.readLine();
    }

    br.close();
    f.close();
}
```

# Handling Exception (2 contd.)

```
public void openFileAndReadLines() throws IOException {
    ...
}

public static void main(String[] args) {
    try {
        openFileAndReadLines();
    }
    catch (IOException e) {
        e.printStackTrace();
    }
}
```

# try ... catch ... finally ...

- The `finally` statement lets you execute code, after `try ... catch` regardless of the result

```
try {
   int[] myNumbers = {1, 2, 3};
   System.out.println(myNumbers[10]);
}
catch (Exception e) {
   System.out.println("Something went wrong.");
}
finally {
   System.out.println("The 'try catch' is finished.");
}
```

UNTAR
Universitas Tarumanagara
UNTAR untuk INDONESIA