Three Types of Errors

- Syntax Errors
    - These are errors that tend to show up and prevent a program from compiling or run in the first place
- Runtime Errors
    - These are errors that tend to show up while a program is running, usually causing it to crash
- Logical Errors
    - These are errors that neither stop a program from compiling nor from crashing, but lead to different outcomes than otherwise expected.

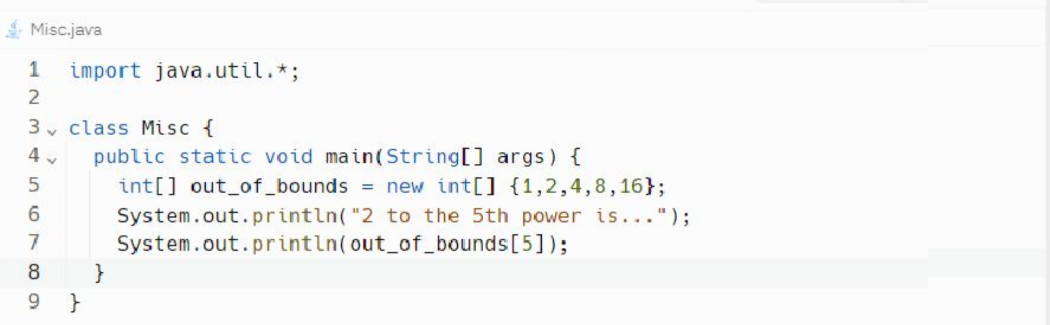
Syntax Errors

- Syntax errors, like the name implies, are errors that that arise from mistakes in following a programming language's syntactic structure.
- Syntax errors are generally detected at compile-time and prevent a program from executing its code.
- Syntax errors often occur due to spelling mistakes and/or inconsistencies (including capitalization) as well as missing, misplaced, and/or extraneous characters such as parentheses (), curly brackets {}, or quotes "".
- IDEs such as Repl.it can be very helpful in catching and correcting syntax errors like those above, but it can't catch everything.
- Other syntax errors are be dependent on the programming language, and what might be a syntax error in one language might be another type of error in another language or potentially not cause an error at all.
- In Java particularly, syntax errors can arise if one is not careful about data types.

Runtime Errors

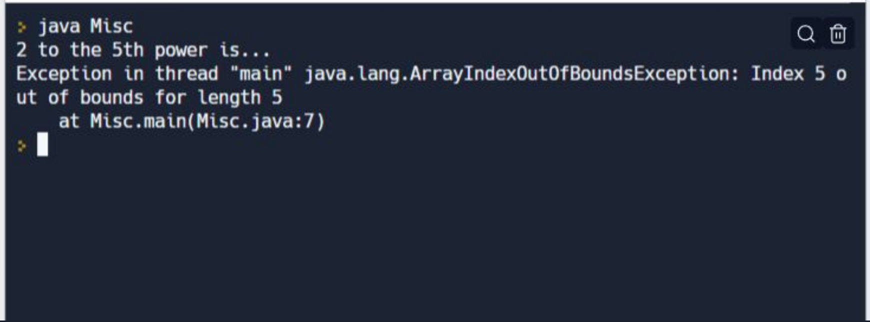
- Even if no errors for a program are detected during compile-time, that doesn’t guarantee that a program is error free.
- Like the name implies, runtime errors are errors that occur while a program is running, generally causing the program to stop and preventing the execution of code after the source of the error.
- While runtime errors can occur for a number of reasons, the slides that follow will showcase some of the most common causes of runtime errors

Runtime Errors (cont.)

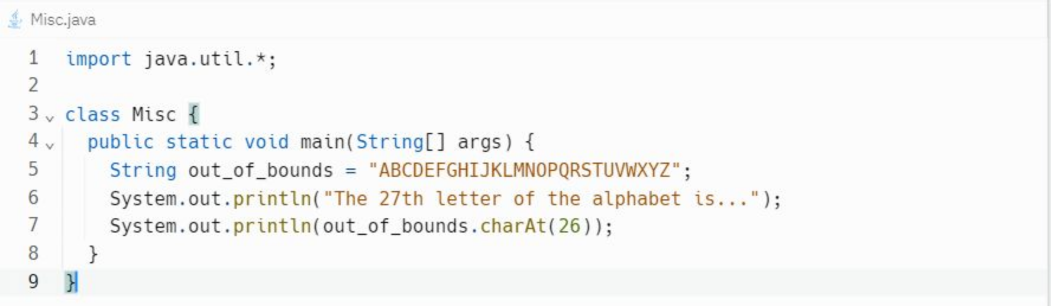


```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    int[] out_of_bounds = new int[] {1,2,4,8,16};
    System.out.println("2 to the 5th power is...");
    System.out.println(out_of_bounds[5]);
  }
}
```

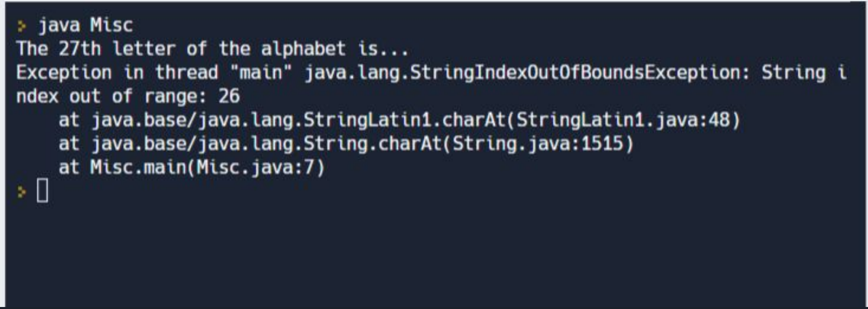


```
> java Misc
2 to the 5th power is...
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 o
ut of bounds for length 5
    at Misc.main(Misc.java:7)
> 
```



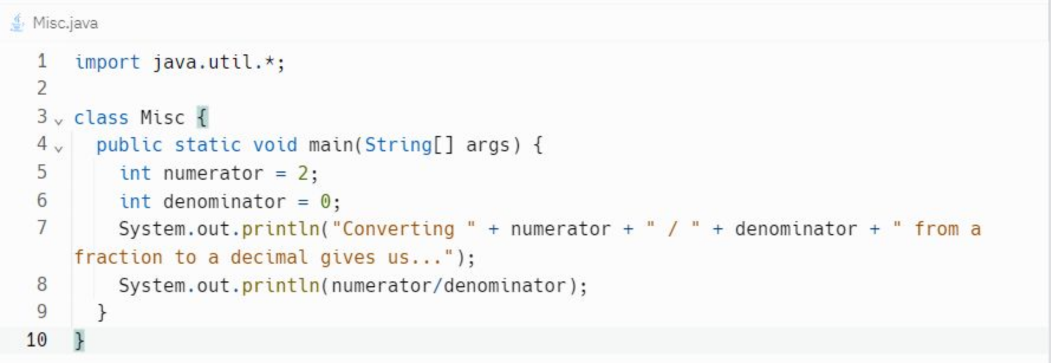
```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    String out_of_bounds = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    System.out.println("The 27th letter of the alphabet is...");
    System.out.println(out_of_bounds.charAt(26));
  }
}
```



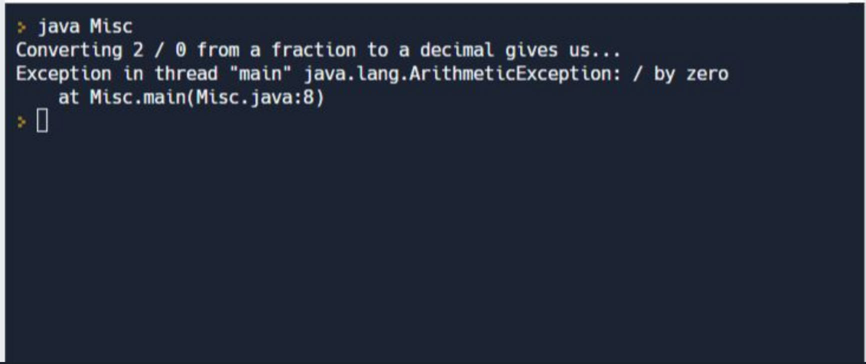
```
> java Misc
The 27th letter of the alphabet is...
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String i
ndex out of range: 26
    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)
    at java.base/java.lang.String.charAt(String.java:1515)
    at Misc.main(Misc.java:7)
> 
```

Runtime Errors (cont.)



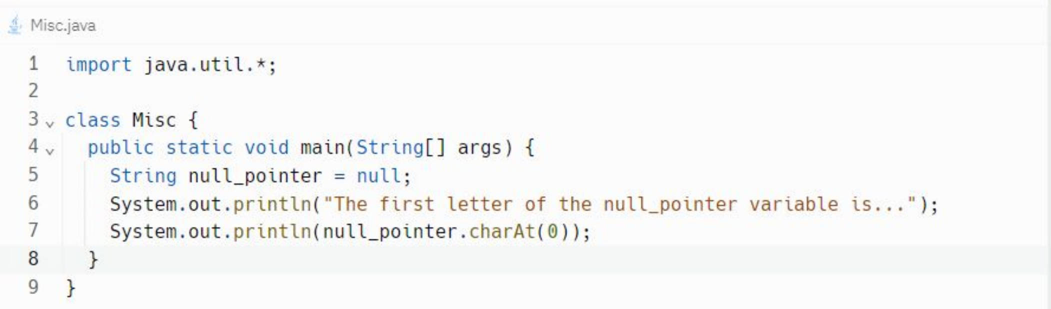
```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    int numerator = 2;
    int denominator = 0;
    System.out.println("Converting " + numerator + " / " + denominator + " from a fraction to a decimal gives us...");
    System.out.println(numerator/denominator);
  }
}
```



```
> java Misc
Converting 2 / 0 from a fraction to a decimal gives us...
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at Misc.main(Misc.java:8)
> 
```

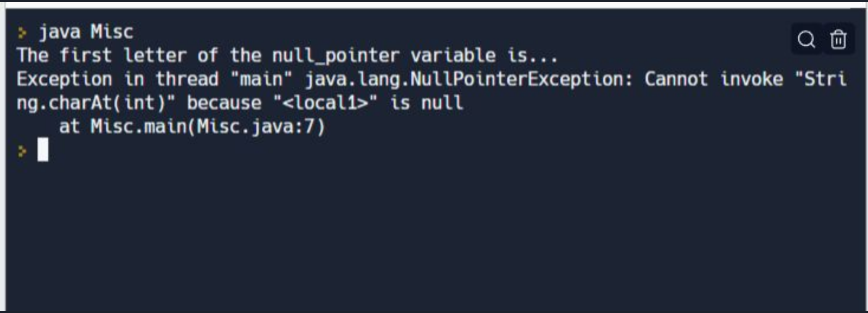
Runtime Errors (cont.)



Misc.java

```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    String null_pointer = null;
    System.out.println("The first letter of the null_pointer variable is...");
    System.out.println(null_pointer.charAt(0));
  }
}
```

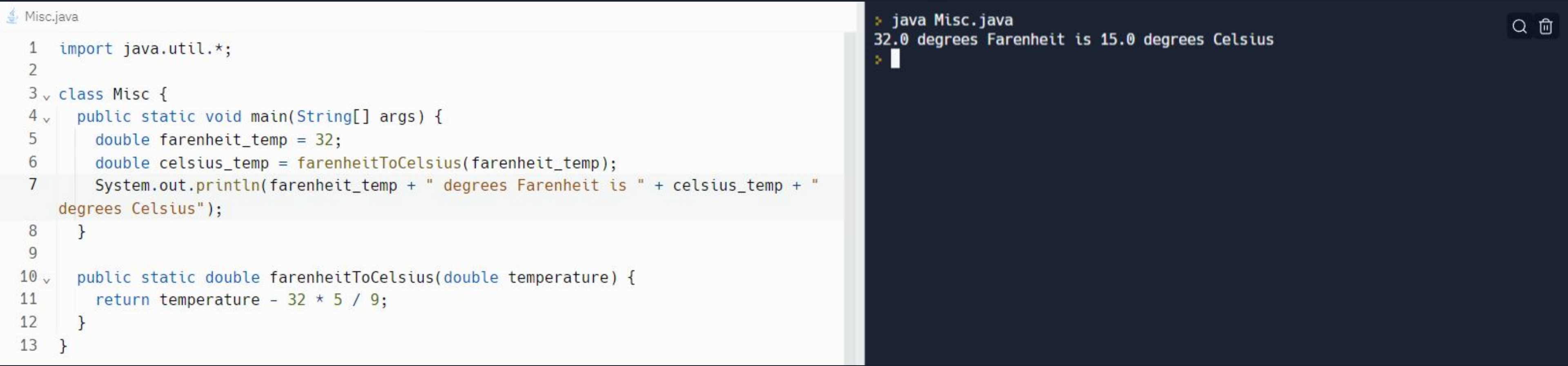


```
> java Misc
The first letter of the null_pointer variable is...
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "Stri
ng.charAt(int)" because "<local1>" is null
    at Misc.main(Misc.java:7)
> 
```

Logic Errors

- Remember that computers will do what we tell them to do which is not always necessarily what we want them to do.
- Logic errors aren’t errors in the same way that syntax errors and runtime errors are. Logic errors refer to instances where a program behaves differently than expected.
- Logic errors can vary greatly from using the wrong variable by accident to having an incorrect algorithm altogether.
- The following slides contain some examples of logic errors although it is by no means an exhaustive list.

Logic Errors (cont.)



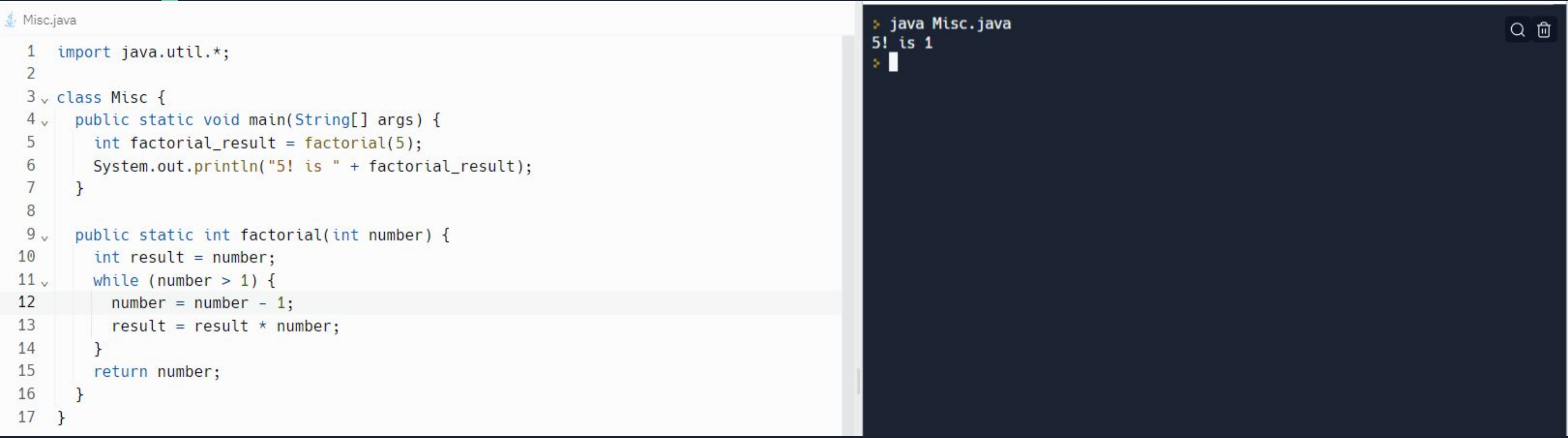
```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    double farenheit_temp = 32;
    double celsius_temp = farenheitToCelsius(farenheit_temp);
    System.out.println(farenheit_temp + " degrees Farenheit is " + celsius_temp + " degrees Celsius");
  }

  public static double farenheitToCelsius(double temperature) {
    return temperature - 32 * 5 / 9;
  }
}
```

```
> java Misc.java
32.0 degrees Farenheit is 15.0 degrees Celsius
>
```

Logic Errors (cont.)



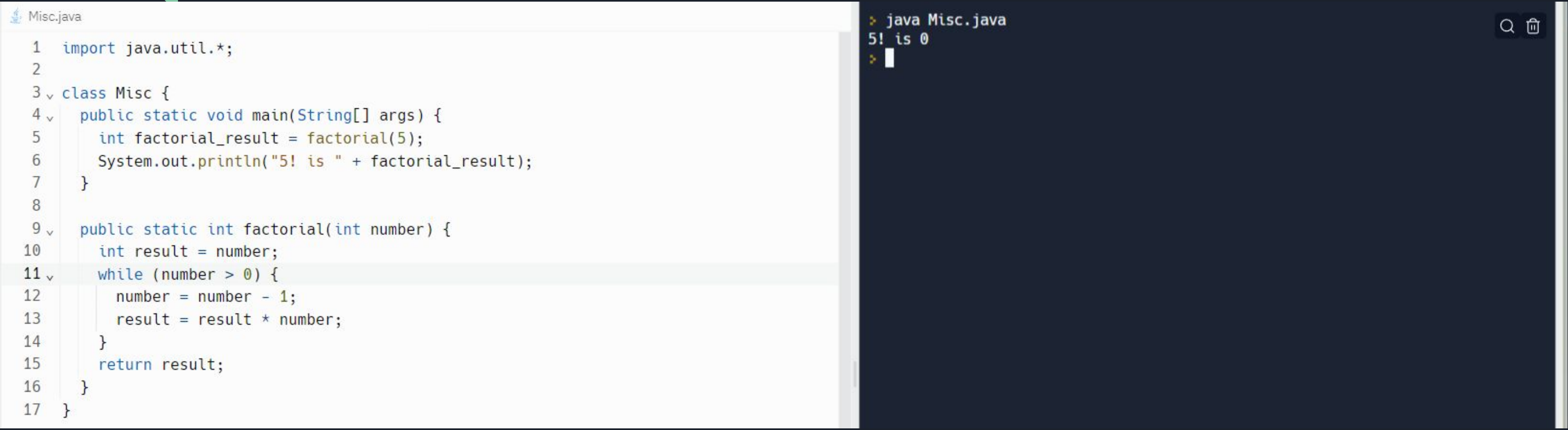
```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    int factorial_result = factorial(5);
    System.out.println("5! is " + factorial_result);
  }

  public static int factorial(int number) {
    int result = number;
    while (number > 1) {
      number = number - 1;
      result = result * number;
    }
    return number;
  }
}
```

```
> java Misc.java
5! is 1
> 
```

Logic Errors (cont.)



```java
import java.util.*;

class Misc {
  public static void main(String[] args) {
    int factorial_result = factorial(5);
    System.out.println("5! is " + factorial_result);
  }

  public static int factorial(int number) {
    int result = number;
    while (number > 0) {
      number = number - 1;
      result = result * number;
    }
    return result;
  }
}
```

```
> java Misc.java
5! is 0
>
```