Java Basics
Some Extra Bits

Input – Output
Formatted Output via "printf()"

- Formatted Output via "printf()"
- print() and println() do not provide output formatting
- printf() for formatted output
- A format specifier begins with a '%' and ends with the conversion code

Input – Output
Formatted Output via "printf()"

- %d for integer
- %f for floating-point number
- %c for character
- %s for string
- Optional [width] can be inserted in between to specify the field-width.
- optional [flags] can be used to control the alignment

Input – Output
Formatted Output via "printf()"

- %αd: integer printed in α spaces (α is optional)
- %αs: String printed in α spaces (α is optional).
- %α.βf: Floating point number (float and double) printed in α spaces with β decimal digits (α and β are optional)
- %n: a system-specific new line (Windows uses "\r\n", Unix and Mac "\n").

Input – Output
Formatted Output via "printf()"

- System.out.printf("Hello%2d and %6s", 8, "HI!!! %n");
- System.out.printf("Hi,%s%4d%n", "Hello", 88);
- System.out.printf("Hi, %d %4.2f%n", 8, 5.556);
- System.out.printf("Hi,%-4s&%6.2f%n", "Hi", 5.5); // '%-ns' for left-align String
- System.out.printf("Hi, Hi, %.4f%n", 5.56);

Input – Output
Input From Keyboard via "Scanner"

- // Construct a Scanner named "in" for scanning System.in (keyboard)

  Scanner in = new Scanner(System.in);
- **in.nextInt();**
- **in.nextDouble();**
- **in.next();**
- **in.close();**

Input – Output
Input from Text File via "Scanner"

- Scanner in = new Scanner(new File("in.txt"));
- int num = in.nextInt();
- in.close();

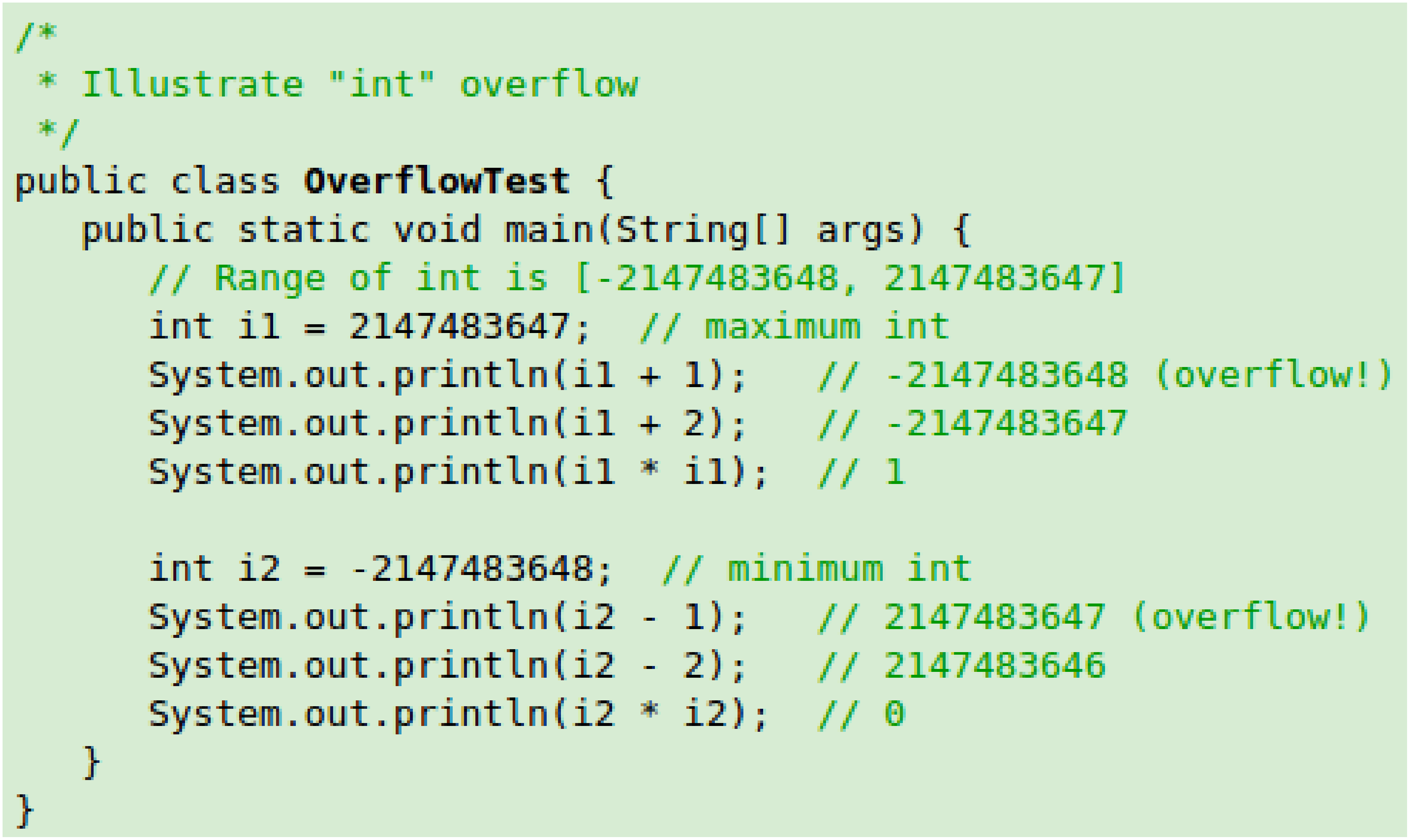
Input – Output
Formatted Output to Text File

- Formatter out = new Formatter(new File("out.txt"));
- out.format("Hi %s,%n", name);
- out.close();

Input – Output
Input via a Dialog Box

- import javax.swing.JOptionPane;
- String radiusStr = JOptionPane.showInputDialog("Enter the radius of the circle");
- radius = Double.parseDouble(radiusStr);

Overflow/Underflow



```java
/*
 * Illustrate "int" overflow
 */
public class OverflowTest {
   public static void main(String[] args) {
      // Range of int is [-2147483648, 2147483647]
      int i1 = 2147483647;  // maximum int
      System.out.println(i1 + 1);   // -2147483648 (overflow!)
      System.out.println(i1 + 2);   // -2147483647
      System.out.println(i1 * i1);  // 1

      int i2 = -2147483648;  // minimum int
      System.out.println(i2 - 1);   // 2147483647 (overflow!)
      System.out.println(i2 - 2);   // 2147483646
      System.out.println(i2 * i2);  // 0
   }
}
```

Overflow/Underflow

- In arithmetic operations, the resultant value wraps around if it exceeds its range (i.e., overflow).
- Java runtime does NOT issue an error/warning message but produces an incorrect result.
- On the other hand, integer division produces an truncated integer and results in so-called underflow.

Overflow/Underflow

- For example, 1/2 gives 0, instead of 0.5.
- Java runtime does NOT issue an error/warning message, but produces an imprecise result.

Conditional Operator (? :)

- A conditional operator is a ternary (3-operand) operator, in the form of booleanExpr ? trueExpr : falseExpr.
- Depending on the booleanExpr, it evaluates and returns the value of trueExpr or falseExpr.
- booleanExpr ? trueExpr : falseExpr

Compound Assignment Operators

- +=
- -=
- *=
- /=
- %=

Exception Handling

-
  ```
  try{

  }catch(Exception ex){

  }finally{

  }
  ```