

Working with Numbers

→ Boxing and unboxing

Stack of integer values

```
final Stack<Integer> values = new Stack<>()  
  
values.push(3);  
values.push(1);  
values.push(10);  
  
while (!values.empty()) {  
    final int i = values.pop();  
    System.out.println(i);  
}
```

```
10  
1  
3
```

- Supports searching of objects based on:
 - `public boolean equals(Object obj)`
 - `public int hashCode()`
- Objects only, no primitive types!

Behind the scenes

```
final Stack<Integer> values =  
    new Stack<>();  
  
values.push(3);  
values.push(1);  
values.push(10);  
  
while (!values.empty()) {  
    System.out.println(values.pop().  
        getClass().getTypeName());  
}
```

```
java.lang.Integer  
java.lang.Integer  
java.lang.Integer
```

Boxing and unboxing

```
int iPrimitive ① = 7;  
  
Integer iInteger = ②  
    iPrimitive;  
  
int iPrimitiveFromInteger = ③  
    iInteger;
```

```
int iPrimitive ① = 7;  
  
Integer iInteger = ②  
    Integer.valueOf(iPrimitive);  
  
int iPrimitiveFromInteger = ③  
    iInteger.intValue();
```

Boxing syntax comparison

```
final Stack<Integer> values  
= new Stack<>();  
  
values.push(Integer.valueOf(3));  
values.push(Integer.valueOf(1));  
values.push(Integer.valueOf(10));  
  
while (!values.empty()) {  
    System.out.println(values.pop().  
        intValue());  
}
```

```
final Stack<Integer> values =  
    new Stack<>();  
  
values.push(3);  
values.push(1);  
values.push(10);  
  
while (!values.empty()) {  
    System.out.println(values.pop())  
}
```

178. Auto boxing int to Double?

Working with Numbers
→ Number Parsing

Parsing `Integer` user input

```
String userInput = null;
try (final Scanner scanner =
    new Scanner(System.in)){
    System.out.print("Enter an integer:");
    userInput = scanner.nextLine();

    final int value = Integer.parseInt(userInput)

    System.out.println("You entered " + value);
} catch (final NumberFormatException e) {
    System.out.println("Sorry, but '" + userInput
        "' is not an integer.");
}
```

Enter an integer:-34
You entered -34

Enter an integer:**five**
Sorry, but '**five**' is
not an integer.

Followup exercise

179. Why using `String userInput = null`?

Parsing binary representation

```
final int value =  
    Integer.parseInt("1101", 2);  
System.out.println("Value: " + value);
```

Value: 13

```
final int value =  
    Integer.parseInt("201", 2);  
System.out.println("Value: " + value);
```

```
Exception in thread "main"  
java.lang.NumberFormatException:  
For input string: "201"  
...  
at de.hdm_stuttgart.sd1...
```

Standard parse methods

- `parseByte()`
- `parseShort()`
- `parseInt()`
- `parseLong()`
- `parseFloat()`
- `parseDouble()`
- `parseBoolean()`

Followup exercises

180. Parsing short values

181. Parsing short values in hexadecimal representation

Working with Numbers

→ Number Formatting

A Locale object represents a specific geographical, political, or cultural region.

An operation that requires a Locale to perform its task is called locale-sensitive and uses the Locale to tailor information for the user.

For example, displaying a number is a locale-sensitive operation: Numbers should be formatted according to the customs and conventions of the user's native country, region, or culture.

Locale properties

- Language
- Encoding
- Country
- Extensible

Get a `NumberFormat` instance

```
final NumberFormat standard = new DecimalFormat();  
System.out.println(standard.format(1234.5678));  
  
final NumberFormat de =  
    DecimalFormat.getInstance(Locale.GERMANY);  
System.out.println(de.format(1234.5678));
```

1234.5678
1.234,568

Create a custom formatter

```
final DecimalFormatSymbols unusualSymbols =  
    new DecimalFormatSymbols(Locale.getDefault());  
unusualSymbols.setDecimalSeparator('|');  
unusualSymbols.setGroupingSeparator('^');  
  
final String strange = "#,##0.###";  
final DecimalFormat weirdFormatter = new DecimalFormat(strange, unusualSymbols);  
weirdFormatter.setGroupingSize(4);  
  
System.out.println(weirdFormatter.format(12345.678));
```

```
1^2345|678
```

182. Locale definitions

183. Formatting int, double and LocalDate

Polymorphic number parsing

```
final NumberFormat de = NumberFormat.getInstance(Locale.GERMANY),
                us = NumberFormat.getInstance(Locale.US);
try {
    final Number[] values = {
        de.parse("103.234"), de.parse("400.000,234"),
        us.parse("103.234"), us.parse("400.000,234"),};
    for (final Number n: values) {
        System.out.println(n + "(" + n.getClass().getTypeName() + ")");
    } catch (ParseException e) { ... }
}
```

```
103234(java.lang.Long)
400000.234(java.lang.Double)
103.234(java.lang.Double)
400(java.lang.Long)
```

Working with Numbers

→ Working with Money

Limited float precision

```
final float result = 0.99f - 0.1f -0.1f -0.1f;  
System.out.println(result);
```

0.68999994

Limited double precision

```
final double result = 0.99 - 0.1 - 0.1 - 0.1;  
System.out.println(result);
```

```
0.6900000000000001
```

Using `BigDecimal`

```
final BigDecimal zero_dot_99 = new BigDecimal("0.99");
final BigDecimal zero_dot_1 = new BigDecimal("0.1");

BigDecimal
    result = zero_dot_99.subtract(zero_dot_1); // Subtracting 0.1

    result = result.subtract(zero_dot_1);      // Subtracting 0.1
    result = result.subtract(zero_dot_1);      // Subtracting 0.1

System.out.println(result);
```

0.69

Chaining `BigDecimal` operations

```
final BigDecimal zero_dot_99 = new BigDecimal("0.99");
final BigDecimal zero_dot_1 = new BigDecimal("0.1");

BigDecimal result = zero_dot_99.
    subtract(zero_dot_1).
    subtract(zero_dot_1).
    subtract(zero_dot_1);

System.out.println(result);
```

0.69

184. Chaining subtract method calls

BigDecimal features

- Higher memory allocation hosting higher precision.
- Immutable instances
- Calculation performance penalty.
- Clumsy interface.

Working with Numbers

⇒ Generating Random Numbers

Using `static double random()`

```
for (int i = 0; i < 10; i++) {  
    System.out.println(Math.random());  
}
```

```
0.4754286988826202  
0.0060114391743414375  
...  
0.9047785351372987  
0.2516070321935864
```

Seeding a pseudo random generator

```
try(final Scanner scanner = new Scanner(System.in)) {  
    System.out.print("Enter an integer seed:");  
    final long seed = scanner.nextLong();  
  
    Random generator = new Random(seed);  
    for (int i = 0; i < 10; i++) {  
        System.out.print(generator.nextBoolean() + " ");  
    }  
}
```

```
Enter an integer seed:4237549835735  
false true true true false false false true false true
```