Formatting Output

Formatting Integers and Decimal Values

NumberFormat

- Used to format numbers using different formats
  - Currency – NumberFormat.getCurrencyInstance()
  - Integers – NumberFormat.getIntegerInstance()
  - Percents – NumberFormat.getPercentInstance()
  - Numbers – NumberFormat.getNumberInstance()
- When using the currency instance you can specify a Locale
  - `Locale locale = Locale.CANADA;`
- Use format() and parse() method to define and display how you will format the different numbers.
  - format() and parse() are opposites
  - format – formats a number using the chosen locale
  - parse – obtains the number from the String based on the locale and format

Using Format

```
Currency:
Locale locale = Locale.GERMANY;
String string = NumberFormat.getCurrencyInstance(locale).format(123.45);
// 123,45 DM

locale = Locale.CANADA;
string = NumberFormat.getCurrencyInstance(locale).format(123.45);
// $123.45

Number:
locale = Locale.GERMAN;
string = NumberFormat.getNumberInstance(locale).format(-1234.56);
// -1.234,56

Percent:

Locale locale = Locale.US;
String string = NumberFormat.getPercentInstance(locale).format(123.45);
// 12,345%
```

Using Parse

```
Currency:
locale = Locale.CANADA;
double num1 = NumberFormat.getCurrencyInstance(locale).parse("$123.45");
// 123.45

Number:
double num1 = NumberFormat.getNumberInstance(locale.GERMAN).parse("-1.234,56");
// 1234.56
```

Percent:

```
double num1 = NumberFormat.getPercentInstance(locale.US).parse("123.45%");
// 1.2345
```

DecimalFormat

- DecimalFormatter objects are obtained using the standard way for obtaining a class (new)
  - DecimalFormat formatter = new DecimalFormat(String);
- Other useful methods are applyPattern and format
  - applyPattern allows you to specify a pattern after you obtain the decimal formatter.
    - formatter.applyPattern(“”) – pattern goes in the double quotes
  - format performs the actual conversion from the double number to a String using the specified pattern.

Symbols used in Patterns

- The 0 symbol shows a digit or 0 if no digit present
- The # symbol shows a digit or nothing if no digit present
- The . symbol indicates the decimal point
- The , symbol is used to group numbers
- The ; symbol is used to specify an alternate pattern for negative values
- The ' symbol is used to quote literal symbols

Examples:

```
DecimalFormat formatter = new DecimalFormat("000000");
String s = formatter.format(-1234.567);  // -001235

formatter = new DecimalFormat("##");
s = formatter.format(-1234.567);         // -1235
s = formatter.format(0);                 // 0


formatter = new DecimalFormat(".00");
s = formatter.format(-.567);             // -.57
formatter = new DecimalFormat("0.00");
s = formatter.format(-.567);             // -0.57



formatter = new DecimalFormat("#,###,###");
s = formatter.format(-1234.567);         // -1,235
s = formatter.format(-1234567.890);      // -1,234,568

formatter = new DecimalFormat("#;(#)");
s = formatter.format(-1234.567);         // (1235)

formatter = new DecimalFormat("'#'#");
s = formatter.format(-1234.567);         // -#1235
formatter = new DecimalFormat("'abc'#");
s = formatter.format(-1234.567);         // -abc1235
```