### Primitive types (2.1)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>integers</td>
<td>42, -3, 92851</td>
</tr>
<tr>
<td>double</td>
<td>real numbers</td>
<td>3.14, 2.0</td>
</tr>
</tbody>
</table>
| char    | a character of text  | 'a', 'X', '
' |
| boolean | logical values       | true, false |

### Expressions (2.1)

- **precedence:** () before */% before +-  
- with int, / is integer quotient and % is integer remainder  
- Strings can be concatenated with other values

```
1 * 2 + 3 * 5 / 4  // "5" + 9.0 / 4.0 + 1
2 + 3 * 5 / 4      // "5" + 2.25 + 1
2 + 15 / 4         // "$2.25" + 1
2 + 3              // "$2.251"
```

### Variables (2.2)

- **type name;** declaration (creates a variable but doesn't give it any value)  
- **name = value;** assignment (stores a value into a variable)  
- **type name = value;** declaration-initialization (creates a variable and stores a value into it)

```
int x;
int y = 3;
x = 1 + y * 2; // x stores the value 7
```

### The for loop (2.3)

- **(repeats a group of statements a fixed number of times)**

```
for (initialization; test; update) {
    statement;
    statement;
    ...
    statement;
}

for (int i = 1; i <= 10; i++) {
    System.out.println(i + " squared is " + (i * i));
}
```

### Nested for loops (2.3)

- **(loops inside loops, can be used to produce complex text patterns)**

```
for (int line = 1; line <= 5; line++) {
    for (int j = 1; j <= (-1 * line + 5); j++) {
        System.out.print(".");
    }
    System.out.println(line);
}
```

### Class constants (2.4)

- **(unchangeable global values that can be seen throughout your program)**

```
public static final type name = value;

public static final int DAYS_PER_WEEK = 7;
```
Parameters (3.1) (A way to pass information in to a method)

Declaration:
```
public static void name(type name, ..., type name) {
    statements;
}
```

Example:
```
public static void box(int width, int height) {
    for (int i = 1; i <= height; i++) {
        for (int i = 1; i <= width; i++) {
            System.out.print("*");
        }
        System.out.println(); // to end the line of output
    }
}
```

Call:
```
methodName(expression, ..., expression);
```

Example:
```
public static void main(String[] args) {
    box(10, 7); // width = 10, height = 7
    box(5, 3); // width = 5, height = 3
}
```

DrawingPanel (3G) (Allows you to draw graphics on a window)
```
import java.awt.
...
DrawingPanel name = new DrawingPanel(width, height);
Graphics g = name.getGraphics();
draw shapes;
```

Example:
```
DrawingPanel panel = new DrawingPanel(400, 300);
Graphics g = panel.getGraphics();
g.drawRect(10, 30, 80, 100);
```

<table>
<thead>
<tr>
<th>Drawing command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>panel.setBackground(color);</td>
<td>sets panel's background color</td>
</tr>
<tr>
<td>g.setColor(color);</td>
<td>sets Graphics pen color (like dipping a brush in paint)</td>
</tr>
<tr>
<td>g.drawLine(x1, y1, x2, y2);</td>
<td>a line from points (x1, y1) to (x2, y2)</td>
</tr>
<tr>
<td>g.drawRect(x, y, width, height);</td>
<td>the outline of rectangle at (x, y)</td>
</tr>
<tr>
<td>g.drawOval(x, y, width, height);</td>
<td>the outline of the largest oval to fit within rectangle of size (width * height) at (x, y)</td>
</tr>
<tr>
<td>g.fillRect(x, y, width, height);</td>
<td>a filled rectangle of size (width x height)</td>
</tr>
<tr>
<td>g.fillOval(x, y, width, height);</td>
<td>a filled oval</td>
</tr>
<tr>
<td>g.drawString(text, x, y);</td>
<td>the given text with its lower-left corner at (x, y)</td>
</tr>
<tr>
<td>g.setFont(font);</td>
<td>sets font to specified font for next strings drawn</td>
</tr>
</tbody>
</table>

Colors and Fonts (3G)
```
new Color(red, green, blue)
new Font(name, style, size)
```

Example:
```
panel.setBackground(Color.YELLOW);
g.setColor(new Color(255, 196, 64));
g.setFont(new Font("Arial", Font.BOLD, 16));
```