CS 312 Parameters

Jeffrey Wang

Using parameters to reduce redundancy

- We started off by hard-coding methods to do one specific thing
- Parameters allow methods to do something infinitely many ways by taking in input

Before parameters

```
public static void print1x() {
     System.out.println("Old MacDonald had a farm,");
     System.out.println("E-I-E-I-O!");
}

public static void print2x() {
     print1x();
     print1x();
}

public static void print4x() {
     print2x();
     print2x();
}
```

After parameters

```
public static void main(String[] args) {
     print(4);
}

public static void print1x() {
     System.out.println("Old MacDonald had a farm,");
     System.out.println("E-I-E-I-O!");
}

public static void print(int n) {
     for (int i = 0; i < n; i++) {
          print1x();
     }
}
```

After parameters, revised

```
public static void main(String[] args) {
     print(4, "Old MacDonald had a farm,\nE-I-E-I-O!");
}

public static void print(int n, String content) {
     for (int i = 0; i < n; i++) {
          System.out.println(content);
     }
}
```

Challenge: draw m*n-sized box

- How could we do this with parameters?
- How many methods do we need now?
- What parameters do we need to draw the box?

Solution

```
public static void main(String[] args) {
     drawBox(10, 20);
}

public static void drawBox(int m, int n) {
     for (int i = 0; i < m; i++) {
          System.out.print("|");
          for (int j = 0; j < n; j++) {
               System.out.print("-");
          }
          System.out.print("|");
     }
}
```

`Graphics` methods

Method name	Description
`g.drawLine(`**x1**, **y1**, **x2**, **y2**`);`	line between points (*x1*, *y1*), (*x2*, *y2*)
`g.drawOval(`**x**, **y**, **width**, **height**`);`	outline largest oval that fits in a box of size *width* * *height* with top-left at (*x*, *y*)
`g.drawRect(`**x**, **y**, **width**, **height**`);`	outline of rectangle of size *width* * *height* with top-left at (*x*, *y*)
`g.drawString(`**text**, **x**, **y**`);`	text with bottom-left at *(x, y)*
`g.fillOval(`**x**, **y**, **width**, **height**`);`	fill largest oval that fits in a box of size *width* * *height* with top-left at (*x*, *y*)
`g.fillRect(`**x**, **y**, **width**, **height**`);`	fill rectangle of size *width* * *height* with top-left at (*x*, *y*)
`g.setColor(`**Color**`);`	set `Graphics` to paint any following shapes in the given color

Challenge: make a Graphics method to draw a circle

- How could we do this with parameters?
- How many methods do we need?
- What parameters do we need to draw the circle?

Solution

```
public static void main(String[] args) {
     final int WIDTH = 400;
     final int HEIGHT = 400;
     DrawingPanel dp = new DrawingPanel(WIDTH, HEIGHT);
     Graphics g = dp.getGraphics();
     g.setColor(Color.CYAN);
     drawCircle(g, 0, 0, 100);
}

public static void drawCircle(Graphics g, int x, int y, int n) {
     g.drawOval(x, y, n, n);
}
```

Further steps

- How could we draw more than one circle at once? Perhaps do so methodically?
- Could we allow this method to do more cool functionality for us by taking in more parameters?

Commenting methods with parameters

```
/**
 * this method draws a square at the given coordinates
 * @param xCoordinate the x coordinate to draw the square
 * @param yCoordinate the y coordinate to draw the square
 */

public static void drawSquare(int xCoordinate, int yCoordinate) {
     //code for drawing the square
}
```