

CSE 143X Section Handout #2 Cheat Sheet

Primitive types (2.1)

(kinds of data that can be used by your programs)

Type	Description	Examples
int	integers	42, -3, 92851
double	real numbers	3.14, 2.0
char	a character of text	'a', 'X', '\n'
boolean	logical values	true, false

Expressions (2.1)

(compute a value using arithmetic operations)

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

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

Arithmetic Operators	
Operator	Meaning
+	addition
-	subtraction, negation
*	multiplication
/	division
%	remainder ("modulus")

Variables (2.2)

(pieces of memory that can store a value of a particular type)

type name;
name = value;

declaration (creates a variable but doesn't give it any 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);
}
```

....1
...2
..3
.4
5

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;
```

CSE 143X Section Handout #2 Cheat Sheet 3

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**);

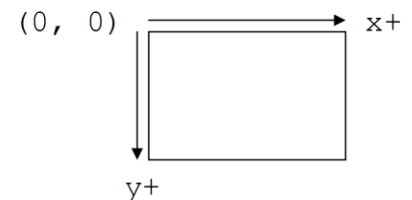
Example:

```
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);
```

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

Colors and Fonts (3G)

Color.BLACK	Color.BLUE	Color.CYAN	Color.DARK GRAY	Color.GRAY
Color.GREEN	Color.LIGHT GRAY	Color.MAGENTA	Color.ORANGE	Color.PINK
Color.RED	Color.WHITE	Color.YELLOW		

```
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));
```