Data types

**Boolean**

boolean – true/false

**Integers**

byte – from -128 to 128

short – from

int – from

long – from

**Floating point**

float – stores an IEEE 32-bit float (decimal numbers)

double – stores an IEEE 64-bit float; stores double the capacity/precision

**Typecasting**

Converting between data types

Widening – converting to data type that can store more data, automatic

Narrowing – converting to data type that can store less data, manual

From narrowest to widest: byte, short, char, int, long, float, double

Use parentheses with the new type for manual typecasting. int a = (int) 6.7;

Operators

**Boolean operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Symbol** | **Description** | **Example** |
| Logical AND | && | Returns true if both inputs are true | true && false = false |
| Logical OR | || | Returns true if at least one of the inputs are true | true || true = true |
| Logical NOT | ! | Negates the input | !true = false |

**Comparison operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Symbol** | **Description** | **Example** |
| Greater than | > | Outputs true if the first input is greater than the second input | 10 > 5 = true |
| Less than | < | Outputs true if the first input is less than the second input | 10 < 5 = false |
| Greater than or equal to | >= | Outputs true if the first input is greater than or equal to the second input | 7 >= 2 = true |
| Less than or equal to | <= | Outputs true if the first input is less than or equal to the second input | 3 <= 3 = true |

**Equality operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Symbol** | **Description** | **Example** |
| Equals | == | Outputs true if the first input is equal to the second input | 4 == 5 is false |
| Not equal | != | Outputs true if the first input is not equal to the second input | 4 != 6 is true |

**Numeric operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Symbol** | **Description** | **Example** |
| Addition | + | Sums the two inputs together | 17 + 4 = 21 |
| Subtraction | - | Subtracts the second input from the first input | 19 - 20 = -1 |
| Multiplication | \* | Multiplies the two inputs together | 7 \* 8 = 56 |
| Division | / | Divides the first input by the second input | 10 / 5 = 2 |
| Modulus | % | Divides the first input by the second input and returns the remainder | 5 % 2 = 1 |

**Augmented operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Symbol** | **Description** | **Example (x = 5)** |
| Addition-assignment | += | Sums the variable on the left with the expression on the right and stores the output back into the variable | x += 8  x is now 13 |
| Subtraction-assignment | -= | Subtracts the expression on the right from the variable on the left and stores the output back into the variable | x -= 3  x is now 2 |
| Multiplication-assignment | \*= | Multiplies the variable on the left with the expression on the right and stores the output back into the variable | x \*= 5  x is now 25 |
| Division-assignment | /= | Divides the variable on the left by the expression on the right and stores the output back into the variable | x /= 5  x is now 1 |
| Modulus-assignment | %= | Divides the variable on the left by the expression on the right and stores the remainder back into the variable | x %= 3  x is now 2 |

Order of Operations

Operators listed from highest to lowest priority

1. Parentheses
2. Post increment/decrement
3. Pre increment/decrement
4. Logical NOT
5. Multiplicative
6. Additive
7. Comparison (excluding equality)
8. Equality
9. Logical AND
10. Logical OR
11. Assignment (includes augmented operators)

If in doubt, use parentheses. It makes programs easier to read and interpret.

Variables

Variables must be declared before they are assigned a variable. Declaration and assignment can be done in one step.

Variable declaration – tells Java the data type and name of the variable you are going to use later

Variable assignment – stores a value in a variable

