# CS180 Lab 02

## Primitive Types & Strings

Week 2
Jan. 14 - Jan. 20

# Types in Java

## Java has eight primitive types:

| | `boolean` | `char` | `byte` | `short` | `int` | `long` | `float` | `double` |
|---|---|---|---|---|---|---|---|---|
| Size | 1 bit | 16 bit | 8 bits | 16 bits | 32 bits | 64 bits | 32 bits | 64 bits |

## `boolean`

- Stores true or false

## `char`

- Stores one ‘character’
  - Basically any key on your keyboard, and then some others
  - Run ‘`man ascii`’ in your terminal
  - Ex: ‘a’, ‘0’, ‘\n’

# `byte`, `short`, `int`, and `long`

- **Stores integer-based numbers**
  - No decimals!
  - Different types based on how large you need the number to be
  - Ex: 451, -219, 0
- **We will mainly use `int`s in CS180**
  - This allows us to use numbers from -2,147,483,648 to 2,147,483,647

# `float` and `double`

- **Stores floating-point (decimal) numbers**
  - Different types based on how large you need the number to be
  - Ex: -891.4312, 3.1415, 0.0
- **We will mainly use `double`s in CS180**
  - You should never have to store a number bigger than this

# What about `Strings`?

- **Strings are not a primitive type in Java, as they are represented using an `Object` instead**
  - Strings are a built-in reference type
  - Ex: “Hello world!”, “Dave”, “0”

# Primitives vs. Reference Types

- **Primitives are pre-defined (i.e. the programmer cannot make more of these)**
- **Reference types can be created by the programmer by creating a `Class`**
  - Reference types are instantiated by creating an **`Object`** of that type
  - **`Objects`** are created by using '**`new`**'

# Common Reference Types

- `String`
  - Store words, sentences, paragraphs, etc.
- `Scanner`
  - Gets input from the user
- `JOptionPane`
  - Allows popup windows to be displayed to the user

# Bitwise Operators

- **All information in a computer is stored in binary (i.e. ‘1’s and ‘0’s)**
  - Bitwise operators allow us to manipulate numbers using binary

| Name | Op | Operation Examples | Binary before operation | Binary after operation (8 bit) | Decimal(s) before | Decimal after |
|---|---|---|---|---|---|---|
| Bit-shift Left | << | 1 << 4 | 00000001 | 00010000 | 1 | 16 |
| Bit-shift Right | >> | 255 >> 6 | 11111111 | 00000011 | 255 | 3 |
| Bitwise AND | & | 170 & 85 | 10101010 & 01010101 | 00000000 | 170, 85 | 0 |
| Bitwise XOR | ^ | 241 ^ 242 | 11110001 ^ 11110010 | 00000011 | 241, 242 | 3 |
| Bitwise OR | \| | 170 \| 85 | 10101010 \| 01010101 | 11111111 | 170, 85 | 255 |

Table created by Purdue Dept. of Computer Science