CSN-103: Fundamentals of Object Oriented Programming

Primitive Types

Java defined 8 primitive types of data:

- byte
- short
- int
- long
- float
- double
- char
- boolean

byte, short, int, long: Integer

float, double: Floating-point

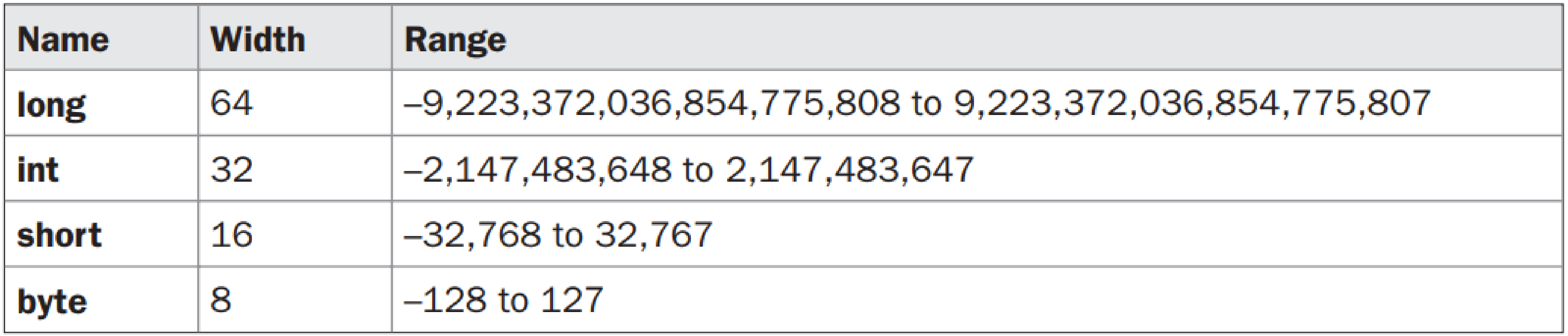
Also referred to as **simple types**

Primitive Types

- Primitive types represent single values (not objects)
- Primitive types have explicit range and mathematical behavior
  - In Java, all data types have strictly defined range
  - This also make Java program "Portable"
  - In C/C++, size of an integer depends on the particular platform

Primitive Types- Integers

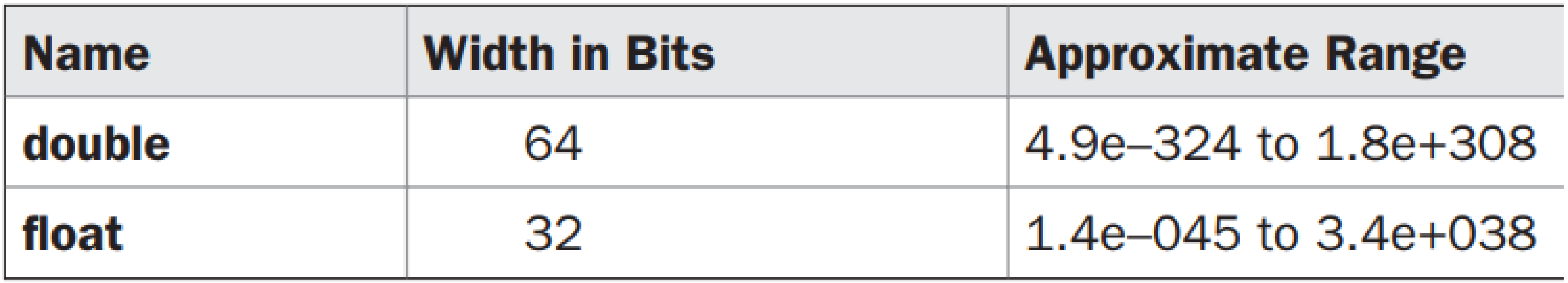
- Java defines four integer types:



Name	Width	Range
long	64	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
int	32	–2,147,483,648 to 2,147,483,647
short	16	–32,768 to 32,767
byte	8	–128 to 127

Primitive Types- Floating Point

- Also known as **real numbers**
- Two kinds of floating point types to store:
  - Single precision
  - Double precision



Name	Width in Bits	Approximate Range
double	64	4.9e–324 to 1.8e+308
float	32	1.4e–045 to 3.4e+038

Primitive Types- Character and Boolean

- In Java, the data type used to store character is **char**
- Java uses Unicode to represent characters
  - Unicode defines fully international character set
    - English, Latin, Greek, and many more
  - Range of char: 0-65536 (16 bits)
  - Also support standard ASCII: 0-127

- Boolean type is used for **logical** values
  - true
  - false

  (1 bit of information but not 1 bit size)

- This is the type returned by relational operators and used by conditional expressions

A Closer look at Constants/Literals

Integer Literals

- Any whole number value is an integer literal
  - Example: 1,2,3,10, 2887…   Decimal values: A base 10 number
- Also possible to use binary, octal, and hexadecimal notation
- Example:

```
int decimal = 495;
int binary = 0b111101111;
int octal = 0757;
int hexa = 0X1EF;
```

Literals- Floating Point

- Represent floating point values with fractional component
- Can be represented as
  - Standard Notation: 3.1234, 56.778
  - Scientific Notation: 6.022E23, 1234E-13, 23e+100
- In Java, floating-point literals are by default **double**

```
double d = 2.335;
```

- To store a literal as **float,** we have to append *F* or *f* to the constant

```
float x = 2.335;                // Error
float x = 2.335f;      // Correct way
```

Boolean Literals

- Used to represent logical values: **true** and **false**
- **true** and **false** do not convert into numerical representation
- true≠1 and false≠0 (Unlike C/C++)
- true and false can be only assigned to boolean variable

Character Literals

- Character in Java are indices of Unicode character set
- 16 bit values → Can be converted into integers and manipulated with integer operators

```
    char ch = 'A';
    ch=ch+1;                    // ch now contains 'B'
```

- Represented within a pair of single quotes

  Visible ASCII characters: 'A' 'k' '6' '@'

  Others: '\t' '\n' '\b'

String Literals

- Sequence of characters enclosed in a pair of double quotes

    "Hello World"

    "These are \n two lines"

    "\" This is shown in Quotes\""

- String in Java is implemented as **object** type, not as array of characters (as in C/C++)