Lecture 3 A
Data Types and Variables

Course: Object Oriented Programming (CS F213)

Prof. Anita Agrawal
BITS Pilani-K.K.Birla Goa campus

Quick recap..

- Software objects are conceptually similar to real-world objects: they too consist of **state** and **related behavior.**
- An object stores its **state** in ***fields*** (variables in some programming languages) and exposes its **behavior** through ***methods*** (functions in some programming languages).
- **Methods** operate on an **object's internal state** and serve as the primary mechanism for object-to-object communication.

Today's session....

- Objects store their state in **fields**. However, the Java programming language uses the term "**variable**" as well.

- In this Lecture……..
  - **Variable naming rules and conventions**
  - **Basic data types**
  - **Default values** and
  - **Literals.**

Types of variables

- 4 types:
  - Instance variables
  - Class variables
  - Local variables
  - Parameters

Types of variables

- **Instance variables (non-static fields):**
  - Objects store their individual states in non-static fields
  - Their values are unique to each instance of a class (to each object, in other words)
- **Example:** the currentSpeed of one bicycle is independent from the currentSpeed of another

Types of variables

- **Class variables (static fields):**
  - This tells the compiler that there is exactly one copy of this variable in existence, regardless of how many times the class has been instantiated.

- **Example:** A field defining the number of legs for a dog could be marked as static since conceptually the same number of legs will apply to all instances.

```
static int legs = 4;
```

Types of Variables

- **Local Variables:**
  - Used by methods.
  - A method will often store its temporary state in local variables.
  - Only visible to the methods in which they are declared; they are not accessible from the rest of the class.
- Example:

```
void calculate()
{
        int count = 0;
….}
```

Local variable

Types of Variables

- **Parameter Variables:**
  - Parameters are always classified as **"variables"** not "fields".
  - This applies to other parameter-accepting constructs as well (such as constructors and exception handlers)
  - Example: The signature for the main method is public static void main(String args []).
    - Here, the **args** variable is the **parameter** to this method.

Primitive Data type

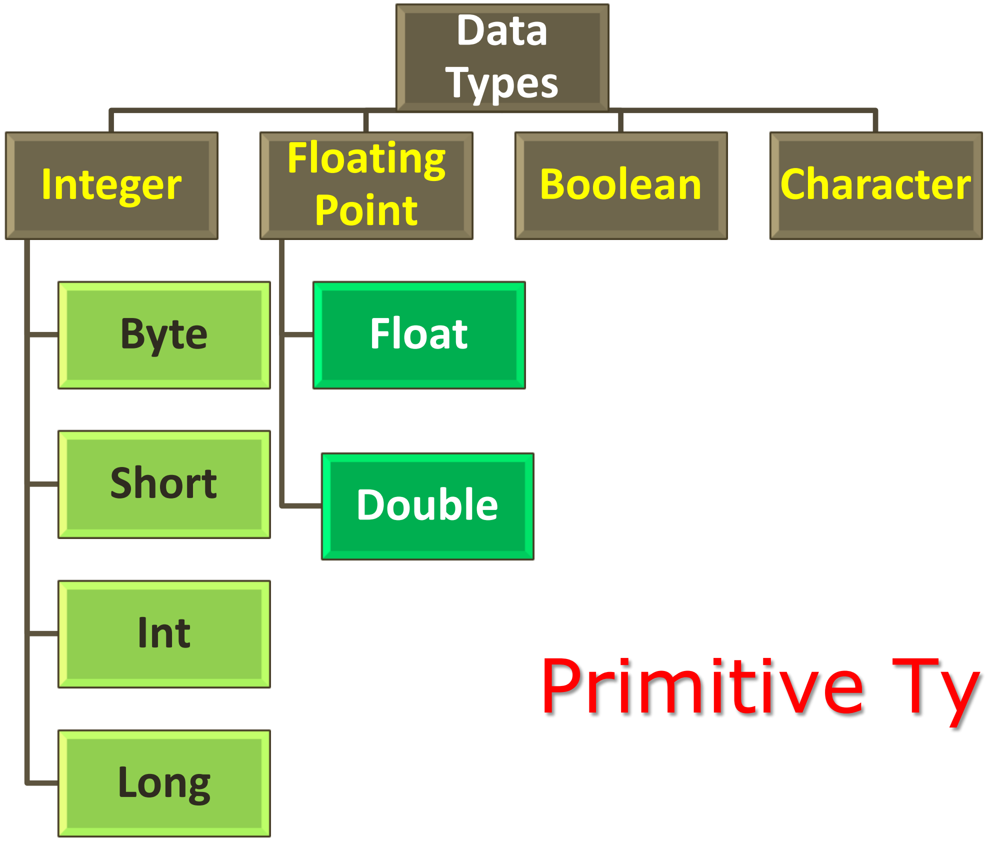
- The Java programming language is **statically-typed**, which means that all variables must first be declared before they can be used. This involves stating the variable's type and name.

- Ex: **int** legs = 2;

- A variable's data type determines the **values** it may contain, plus the **operations** that may be performed on it.

- Primitive types are special data types built into the language; they are not objects created from a class.
- A **primitive type** is predefined by the language and is named by a **reserved keyword**.
- There are **eight** primitives in Java.
- All types have **strictly defined range**
- This make Java program **“Portable”**

- –In C/C++, **size** of an integer depends on the **particular architecture**



Primitive Types

Primitive Data Types in Java: Integers

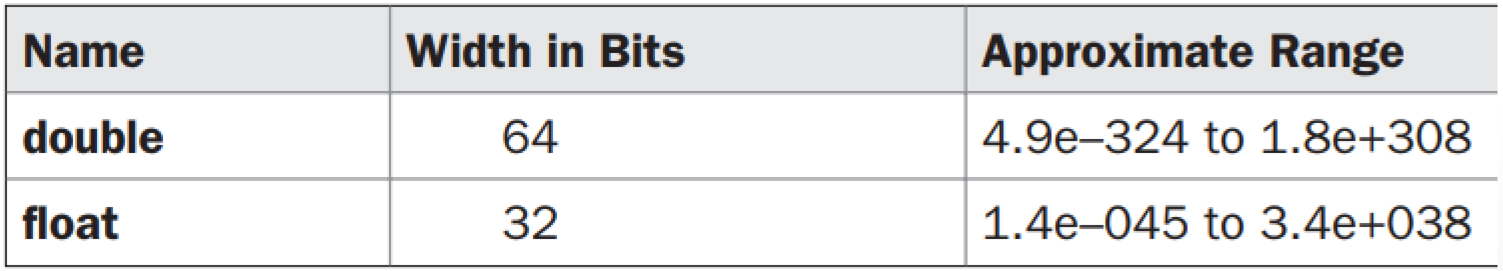
Name	Width	Range
long	64	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
int	32	–2,147,483,648 to 2,147,483,647
short	16	–32,768 to 32,767
byte	8	–128 to 127

- All of the above are stored as signed two’s complement numbers
- Java does not support unsigned positive only integers

Primitive data types in Java: floating-point numbers

Also known as real numbers • Two kinds of floating point types to store: –

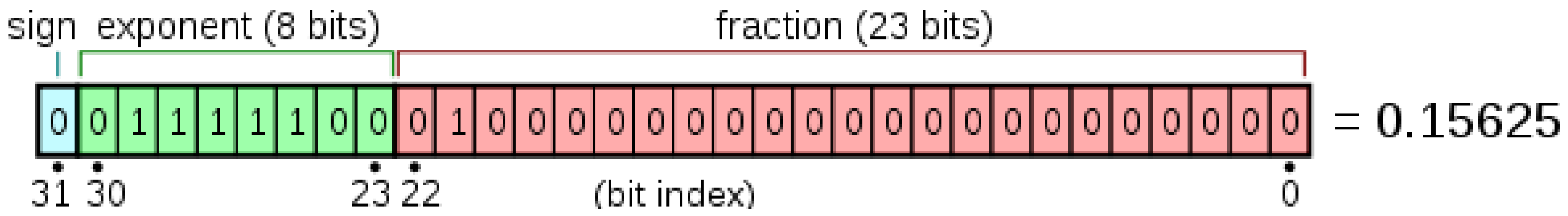
- Float type (Single precision)
- Double Type (Double precision)



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

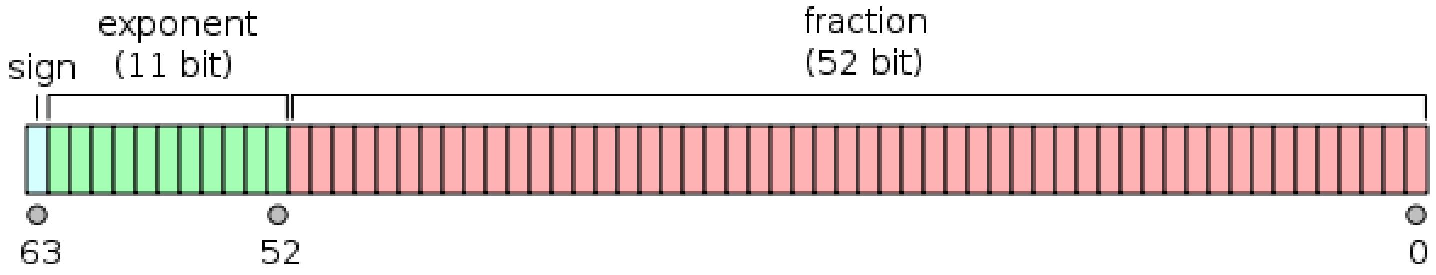
Primitive Data Types in Java: floating point numbers

- **IEEE 754 single-precision binary floating-point format:**
- Sign bit: 1 bit
- Exponent width: 8 bits
- Significand precision: 23 bits (explicitly stored)



Primitive Data Types in Java: floating point numbers

- **IEEE 754 double-precision binary floating-point format:**
- Sign bit: 1 bit
- Exponent width: 11 bits
- Significand precision: 52 bits (explicitly stored)



Java – C: A comparison

- Java is a **strongly** typed language.
- C is a **weakly** typed language

Example: (In C…)

```
int main()
 {
        int i= 3.0;
        printf("Value of i is %d", i);
        return 0;
}
```

Output: Value of i is 3

- Example: (In Java)

```
class Example
{
        public static void main(String args[])
                {
                   int i= 3.0;
                    System.out.println("value of i: "+i);
                }
 }
```

Type mismatch throws an error. (fldemo)

```
error: incompatible types:
     Possible lossy conversion from double to int
            int i=3.0;
```

Char type

- Java uses Unicode to represent characters.
- Unicode defines fully international character set: English, Latin, Greek, and many more
- The minimum value of char data type is '\u0000' (0). The maximum value of char data type is '\uffff'.
- It supports ASCII values also 0-255

Boolean Type

- Takes two values- true and false
- Returned by relational operators and used by conditional expressions

String type

- In addition to the 8 primitive data types, the Java programming language provides special support for character strings via the java.lang.String class.
- It is technically not a primitive data type.
- String objects are immutable.

Literals

Literals

- A ***literal*** is the source code representation of a fixed value
- They are represented directly in your code without requiring computation.
- it's possible to assign a literal to a variable of a primitive type.

Literals-Integer representation

- An integer literal is of type **long** if it ends with the letter L or l; otherwise it is of type **int**
- Integer literals can be expressed by the following number systems:
  - Decimal
  - Octal
  - Hexadecimal
  - Binary
- For general purpose-programming, we use the decimal number system

Syntax to be followed for other number systems

- int decval: 1238 //number in decimal
- int Octval: 01237 //number in octal
- int Hexval: 0x12ABC //number in hexadec
- int binval: 0b10110 //number in binary

Literals: Floating point

- Floating-point literals are written with a decimal point.
- Can be represented as
  - –Standard Notation: 3.1234, 56.778
  - –Scientific Notation: 6.022E23, 1234E-13, 23e+100

- Floating-point literals are by default **double**
  - For example, 5.0 is considered a double value, not a float value.

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

- **float f = 3.145; // Error**
- **float f = 3.145f;// 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