Java Review

Input & Variables

Declaring Variables

Format	Example
`<type> <name>;`	`int number;`

- Variable names, class names, and method names created by a programmer are **`identifiers`**.
- An identifier must begin with a letter and contain only letters, numbers, and some special characters like an underscore ("_").
- Identifiers must not have white spaces. White space is defined as a space, tab or Enter character.
- Java is case sensitive so upper case and lower case letters are different characters.

Primitive Data Types

Type	Values	Default	Size	Range
byte	signed integers	0	8 bits	-128 to 127
short	signed integers	0	16 bits	-32768 to 32767
int	signed integers	0	32 bits	-2147483648 to 2147483647
long	signed integers	0	64 bits	-9223372036854775808 to 9223372036854775807
float	IEEE 754 floating point	0.0	32 bits	+/-1.4E-45 to +/-3.4028235E+38,
double	IEEE 754 floating point	0.0	64 bits	+/-4.9E-324 to +/-1.7976931348623157E+308,
char	Unicode character	\u0000	16 bits	\u0000 to \uFFFF
boolean	true / false	false	1 bit used in 32 bit integer	

Abstract Data Types

- Variables can also be declared using an abstract data type.
- One kind of abstract data type is the **class**.
- A class defines, not just a single piece of data (like a primitive data type). It also includes **methods** for performing actions on the data.

- **`String` is a class; not a primitive data type**
- **Strings in Java are immutable**
  more about this in chapter 6

Notes on Casting

- ***Type Casting***: converting data from one type to another compatible type.
- Some automatic conversions are permitted because they are safe. A `float`, for example, can be stored in a `double` without a cast.
- Converting data **implicitly** (i.e. without an **explicit** cast) is called a *widening* conversion
- This table summarizes the widening conversions or implicit casts:

From	To
byte	**short, int, long, float, double**
short	**int, long, float, double**
char	**int, long, float, double**
int	**long, float, double**
long	**float, double**
float	**double**

- ***Examples:***

```
byte a = 1;
byte b = 2;
byte c = a + b;         // a and b are promoted to int
byte c = (byte)(a + b); // compiles ok

char ch;
int  n;
ch = (char)65;  // decimal value of a space
n = ch;         // implicit cast
ch = (char)n;   // explicit cast
```

- ***Beware:*** implicit casting on literals may cause errors

```
// Fahrenheit to Celsius conversion formula:
double  f = 86.0;
double  c;
c = (f – 32) * 5 / 9;      // always calculates to zero
     // correct version:
c = (f – 32) * 5.0 / 9.0;
```

- ***Truncate:*** casting a double to an in removes (truncates) the decimal place

```
double d = 4.7;
int    n;
n = (int) d;          // n now is 4
```

- What will this print? Why?

```
short s = (short) 75000;
System.out.println(s);
```