جـامعة بيروت العربية
BEIRUT ARAB UNIVERSITY

CMPS 241
Introduction to Programming

Primitive Data Types, Expressions, Variables

Increment and decrement

shortcuts to increase or decrease a variable's value by 1 using unary operators (++ and --)

Shorthand	Equivalent longer version
variable`++;`	**variable** `= ` **variable** `+ 1;`
variable`--;`	**variable** `= ` **variable** `- 1;`

```
int x = 2;
x++;                    // x = x + 1;
                        // x now stores 3

double gpa = 2.5;
gpa--;                  // gpa = gpa - 1;
                        // gpa now stores 1.5
```

Modify-and-assign

shortcuts to modify a variable's value

Shorthand	Equivalent longer version
variable **+=** **value**;	**variable** = **variable** + **value**;
variable **-=** **value**;	**variable** = **variable** - **value**;
variable ***=** **value**;	**variable** = **variable** * **value**;
variable **/=** **value**;	**variable** = **variable** / **value**;
variable **%=** **value**;	**variable** = **variable** % **value**;

```
x += 3;                // x = x + 3;
gpa -= 0.5;            // gpa = gpa - 0.5;
number *= 2;           // number = number * 2;
```

Java Operator Precedence

Description	Operators
Unary Operators	++, --, +, - **Highest**
Binary Multiplicative Operators	*, /, %
Binary Additive Operators	+, -
Assignment Operators	=, +=, -=, *=, /=, %= **Lowest**

- Binary Operators in the same level (such as + and -) are of equal priority and are evaluated left to right. (Example: x * y / 3)
- Unary Operators in the same level (such as + and -) are of equal priority and are evaluated right to left. (Example: ++x - ++y)
- Assignment Operators in the same level (such as =) are of equal priority and are evaluated right to left. (Example: x=y=z=9;)

Example: Evaluate the expression

```
      z – (a + b / 2) + w * -y
      Given  z = 8, a = 3, b = 9, w = 2, y =
-5
            8 – (3 + 9 / 2) + 2 * - -5
        (Step-1)  9/2 = 4
             8 – (3 + 4) + 2 * - -5
        (Step-2)  (3+4) = 7
             8 – 7 + 2 * - -5
        (Step-3)  - - 5 = 5
             8 – 7 + 2 * 5
        (Step-4)  2 * 5 = 10
             8 – 7 + 10
        (Step-5)  8 – 7 = 1
             1 + 10
        (Step-6)  1 + 10 = 11
             11
```

Receipt question

Improve the receipt program using variables.

```java
public class Receipt {
    public static void main(String[] args) {
        // Calculate total owed, assuming 8% tax / 15% tip
        System.out.println("Subtotal:");
        System.out.println(38 + 40 + 30);
        System.out.println("Tax:");
        System.out.println((38 + 40 + 30) * .08);
        System.out.println("Tip:");
        System.out.println((38 + 40 + 30) * .15);
        System.out.println("Total:");
        System.out.println(38 + 40 + 30 +
                           (38 + 40 + 30) * .08 +
                           (38 + 40 + 30) * .15);
    }
}
```

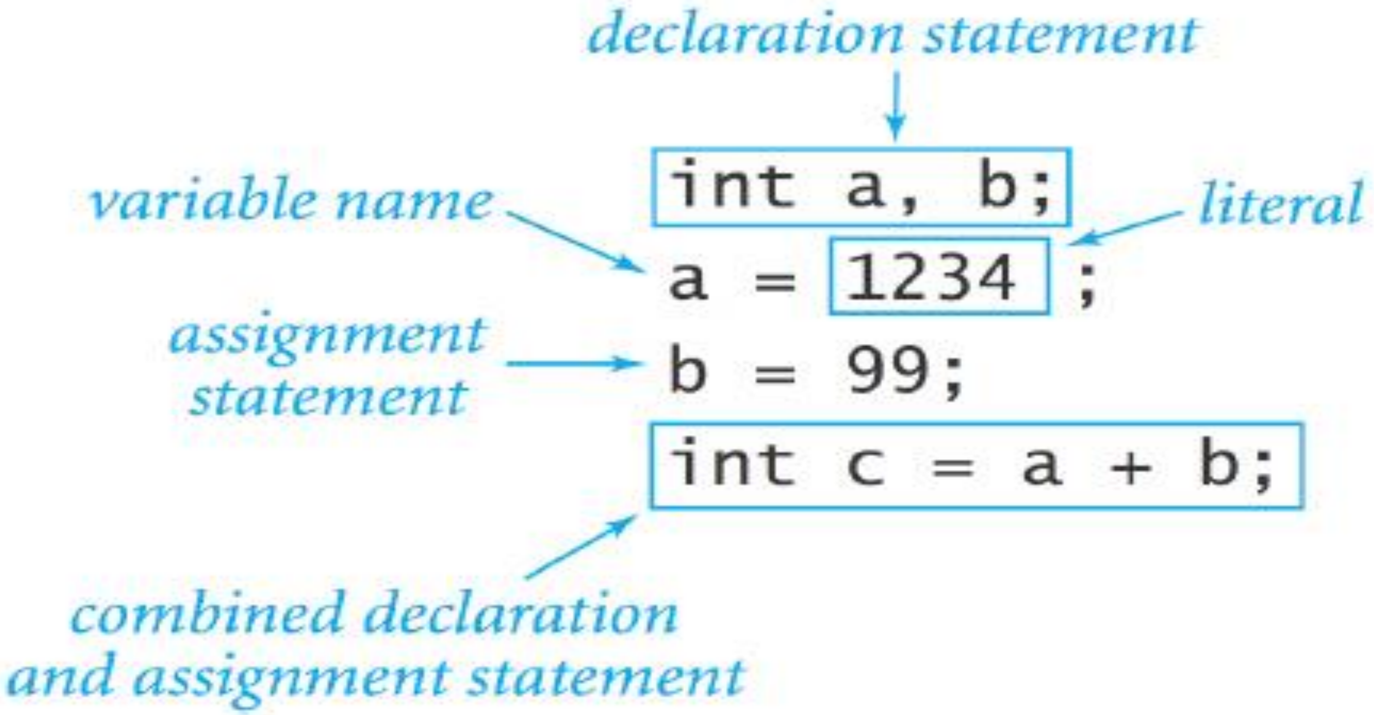
Receipt answer

```
public class Receipt {
    public static void main(String[] args) {
        // Calculate total owed, assuming 8% tax / 15% tip
        int subtotal = 38 + 40 + 30;
        double tax = subtotal * .08;
        double tip = subtotal * .15;
        double total = subtotal + tax + tip;

        System.out.println("Subtotal: " + subtotal);
        System.out.println("Tax: " + tax);
        System.out.println("Tip: " + tip);
        System.out.println("Total: " + total);
    }
}
```

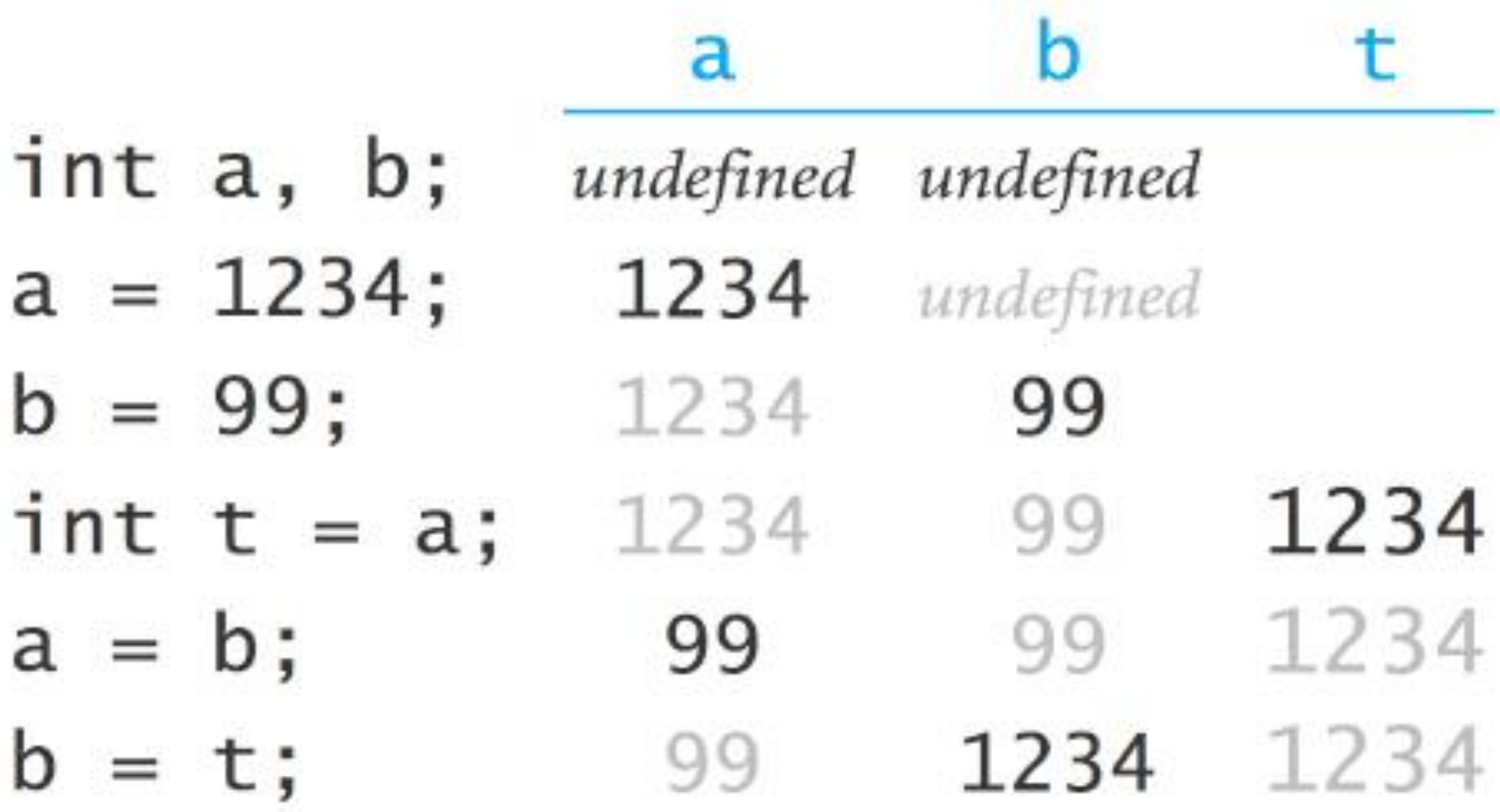
Variables (Summary)

- name, type, value
- declaration and assignment



```
int a, b;
a = 1234 ;
b = 99;
int c = a + b;
```

Trace



	a	b	t
`int a, b;`	*undefined*	*undefined*	
`a = 1234;`	1234	*undefined*	
`b = 99;`	1234	99	
`int t = a;`	1234	99	1234
`a = b;`	99	99	1234
`b = t;`	99	1234	1234

Type casting

- **Type Cast**: A conversion from one type to another.
  - To promote an `int` into a `double` to get exact division from `/`
  - To truncate a `double` from a real number to an integer

- Syntax:

  (**type**) **expression**

  Examples:

```
double result = (double) 19 / 5;      // 3.8
int result2 = (int) result;           // 3
```

More about type casting

- Type casting has high precedence and only casts the item immediately next to it.
  - double x = **(double)** 1 + 1 / 2; **// 1.0**
  - double y = 1 + **(double)** 1 / 2; **// 1.5**

- You can use parentheses to force evaluation order.
  - double average = **(double)** (a + b + c) / 3;

- A conversion to `double` can be achieved in other ways.
  - double average = 1.0 * (a + b + c) / 3;

Examples (Type Casting)

(int)4.8 has value **4**

(double)5 has value **5.0**

(double)(7/4) has value **1.0**

(double)7 / (double)4 has value **1.75**

char data type

- **char** : A primitive data type representing single characters of text (e.g., 'a', 'b', '@', ' ', etc.).

```
public static void main(String[] args) {
    char a = 's';
    System.out.println ("student" + a);
}
```

Output:

```
students
```

Java's primitive types

- **primitive types**: there are 8 simple types for numbers, text, etc.

Type	Description	Size
`int`	The integer type, with range -2,147,483,648 . . . 2,147,483,647	4 bytes
`byte`	The type describing a single byte, with range -128 . . . 127	1 byte
`short`	The short integer type, with range -32768 . . . 32767	2 bytes
`long`	The long integer type, with range -9,223,372,036,854,775,808 . . . -9,223,372,036,854,775,807	8 bytes
`double`	The double-precision floating-point type, with a range of about $\pm 10^{308}$ and about 15 significant decimal digits	8 bytes
`float`	The single-precision floating-point type, with a range of about $\pm 10^{38}$ and about 7 significant decimal digits	4 bytes
`char`	The character type, representing code units in the Unicode encoding scheme	2 bytes
`boolean`	The type with the two truth values `false` and `true`	1 bit

- ***Java*** also has ***object*** types **(e.g. Strings)**, which we'll talk about later