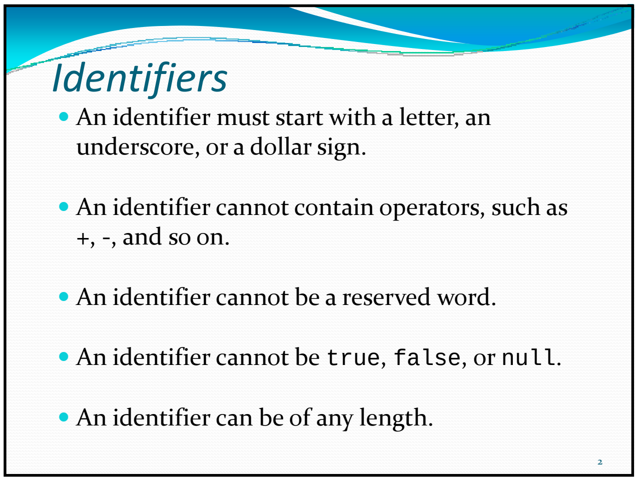
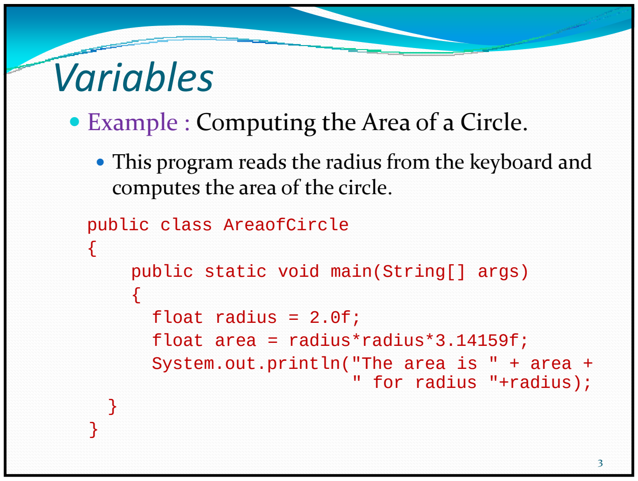


Identifiers

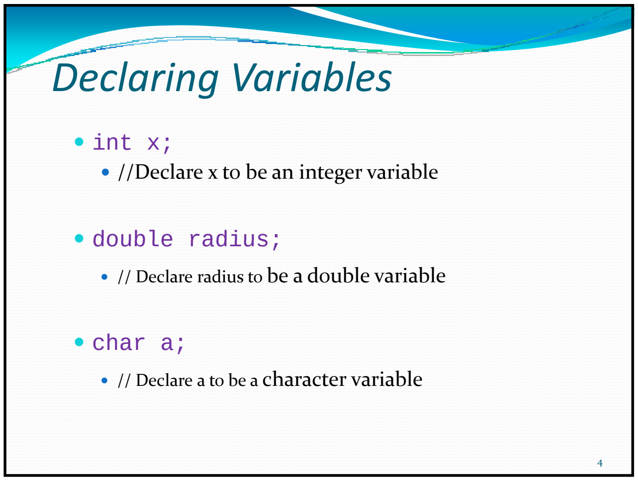
- An identifier must start with a letter, an underscore, or a dollar sign.
- An identifier cannot contain operators, such as +, -, and so on.
- An identifier cannot be a reserved word.
- An identifier cannot be `true`, `false`, or `null`.
- An identifier can be of any length.



Variables

- Example : Computing the Area of a Circle.
  - This program reads the radius from the keyboard and computes the area of the circle.

```
public class AreaofCircle
{
    public static void main(String[] args)
    {
      float radius = 2.0f;
      float area = radius*radius*3.14159f;
      System.out.println("The area is " + area +
                         " for radius "+radius);
  }
}
```



Declaring Variables

- `int x;`
  - //Declare x to be an integer variable
- `double radius;`
  - // Declare radius to be a double variable
- `char a;`
  - // Declare a to be a character variable

Assignment Statements

- `x = 1;`
  - `// Assign 1 to x;`
- `radius = 1.0;`
  - `// Assign 1.0 to radius;`
- `a = 'A';`
  - `// Assign 'A' to a;`

Declaring and Initializing in One Step

- `int x = 1;`
- `double d = 1.4;`
- `float f = 1.4;`
  - Is this statement correct?

Constants

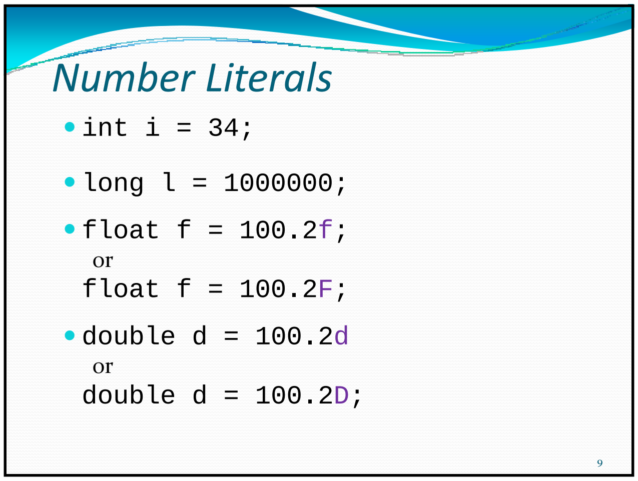
```
final datatype CONSTANTNAME = VALUE;

final double PI = 3.14159;

final int SIZE = 3;
```

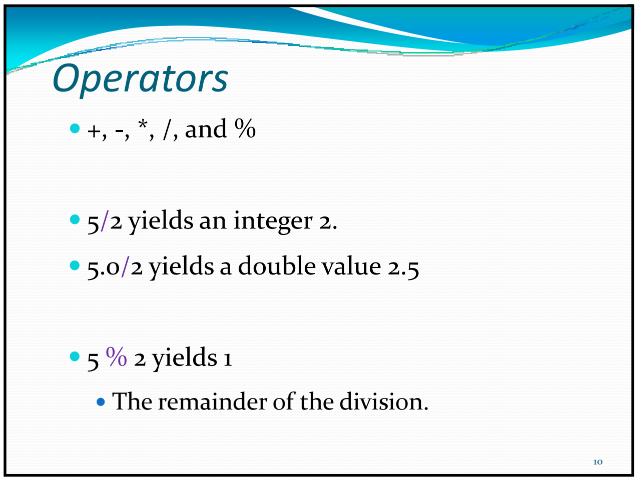
Numerical Data Types

`byte`	8 bits
`short`	16 bits
`int`	32 bits
`long`	64 bits
`float`	32 bits
`double`	64 bits



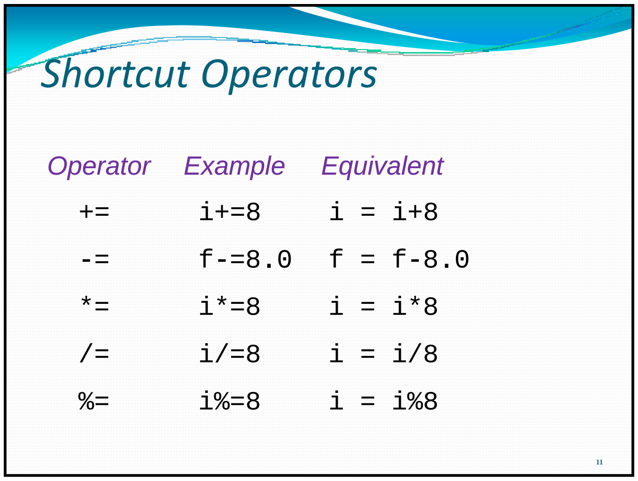
Number Literals

- `int i = 34;`
- `long l = 1000000;`
- `float f = 100.2f;`
  or
  `float f = 100.2F;`
- `double d = 100.2d`
  or
  `double d = 100.2D;`



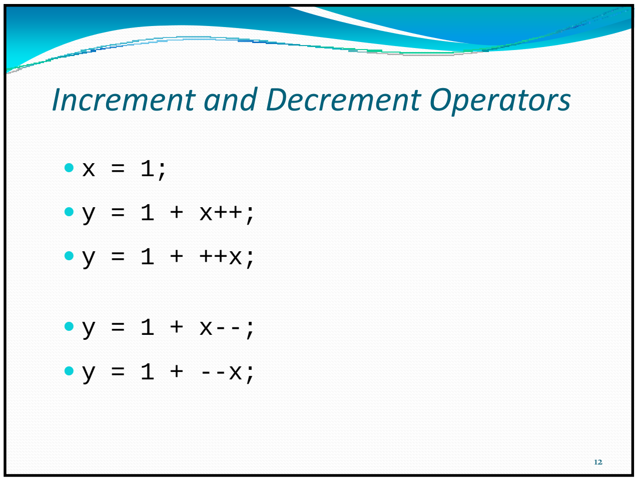
Operators

- +, -, *, /, and %
- 5/2 yields an integer 2.
- 5.0/2 yields a double value 2.5
- 5 % 2 yields 1
  - The remainder of the division.



Shortcut Operators

Operator	*Example*	*Equivalent*
`+=`	`i+=8`	`i = i+8`
`-=`	`f-=8.0`	`f = f-8.0`
`*=`	`i*=8`	`i = i*8`
`/=`	`i/=8`	`i = i/8`
`%=`	`i%=8`	`i = i%8`



Increment and Decrement Operators

- `x = 1;`
- `y = 1 + x++;`
- `y = 1 + ++x;`
- `y = 1 + x--;`
- `y = 1 + --x;`

Numeric Type Conversion

- Consider the following statements:

```
byte i = 100;
long l = i*3+4;
double d = i*3.1+l/2;

int x = l; //(Wrong)
long l = x;
```

  - `//(fine, implicit casting)`

Type Casting

- double
- float
- long
- int
- short
- byte

Type Casting, cont.

- Implicit casting
  - `double d = 3; //(type widening)`
- Explicit casting
  - `int i = (int)3.0; //(type narrowing)`
- *What is wrong?*
  - *int x = 5/2.0;*

Character Data Type

- char letter = 'A'; (ASCII)
- char numChar = '4'; (ASCII)
- char letter = '\u000A'; (Unicode)
- char letter = '\u000A'; (Unicode)

- Special characters
- char tab = ‘\t’;

Unicode Format

Description	Escape Sequence	Unicode
Backspace	\b	\u0008
Tab	\t	\u0009
Linefeed	\n	\u000a
Carriage return	\r	\u000d

The `boolean` Type and Operators

- `boolean lightsOn = true;`
- `boolean lightsOn = false;`

&& (and)	(1 < x) && (x < 100)
`||` (or)	(lightsOn) || (isDayTime)
! (not)	!(isStopped)

The & and | Operators

- If x is 1, what is x after this expression?
  - `(1 > x) & ( 1 > x++)`
- If x is 1, what is x after this expression?
  - `(1 > x) && ( 1 > x++)`
- How about
  - (1 = x) | (1 > x++)?
  - (1 = x) || (1 > x++)?

Operator Precedence

- Casting
- ++,--(preincrement,predecrement)
- *, /, %
- +, -
- <, <=, >, >=
- ==, !=;
- &&, &
- ||, |
- =, +=, -=, *=, /=, %=

Programming Style and Documentation

- Appropriate Comments
- Naming Conventions
- Proper Indentation and Spacing Lines
- Block Styles

Appropriate Comments

- Include a summary at the beginning of the program to explain what the program does, its key features, its supporting data structures, and any unique techniques it uses.
- Include your name, class section, instruction, date, and a brief description at the beginning of the program.

Naming Conventions

- Choose meaning and descriptive names.
- Variables and method names:
  - Use lowercase. If the name consists of several words, concatenate all in one, use lowercase for the first word, and capitalize the first letter of each subsequent word in the name.
  - For example, the variables `radius` and `area`, and the method `computeArea`.

Naming Conventions (cont...)

- Class names
  - Capitalize the first letter of each word in the name. For example, the class name `ComputeArea`.
- Constants
  - Capitalize all letters in constants.
    - For example, the constant `PI`.

Proper Indentation and Spacing

- Indentation
  - Indent two spaces.
- Spacing
  - Use blank line to separate segments of the code.

Block Styles

- Use next-line style for braces.

Programming Errors

- Syntax Errors
  - Detected by the compiler
- Runtime Errors
  - Causes the program to abort
- Logic Errors
  - Produces incorrect result

The End

Questions?