

## Naming

Every programming language has its own set of rules and conventions for the kinds of names that you're allowed to use, and the Java programming language is no different. The rules and conventions for naming your variables can be summarized as follows:

- Variable names are case-sensitive. A variable's name can be any legal identifier — an unlimited-length sequence of Unicode letters and digits, beginning with a letter, the dollar sign \$, or the underscore character "\_". The convention, however, is to always begin your variable names with a letter, not "\$" or "\_". Additionally, the dollar sign character, by convention, is never used at all. You may find some situations where auto-generated names will contain the dollar sign, but your variable names should always avoid using it. A similar convention exists for the underscore character; while it's technically legal to begin your variable's name with "\_", this practice is discouraged. White space is not permitted.
- Subsequent characters may be letters, digits, dollar signs, or underscore characters. Conventions (and common sense) apply to this rule as well. When choosing a name for your variables, use full words instead of cryptic abbreviations. Doing so will make your code easier to read and understand. In many cases it will also make your code self-documenting; fields named `cadence`, `speed`, and `gear`, for example, are much more intuitive than abbreviated versions, such as `s`, `c`, and `g`. Also keep in mind that the name you choose must not be a [keyword or reserved word](#).
- If the name you choose consists of only one word, spell that word in all lowercase letters. If it consists of more than one word, capitalize the first letter of each subsequent word. The names `gearRatio` and `currentGear` are prime examples of this convention. If your variable stores a constant value, such as `static final int NUM_GEAR = 6`, the convention changes slightly, capitalizing every letter and separating subsequent words with the underscore character. By convention, the underscore character is never used elsewhere.

## Java Language Keywords

Here's a list of keywords in the Java programming language. You cannot use any of the following as identifiers in your programs. The keywords `const` and `goto` are reserved, even though they are not currently used. `true`, `false`, and `null` might seem like keywords, but they are actually literals; you cannot use them as identifiers in your programs.

<code>abstract</code>	<code>continue</code>	<code>for</code>	<code>new</code>	<code>switch</code>
<code>assert</code> ***	<code>default</code>	<code>goto</code> *	<code>package</code>	<code>synchronized</code>
<code>boolean</code>	<code>do</code>	<code>if</code>	<code>private</code>	<code>this</code>
<code>break</code>	<code>double</code>	<code>implements</code>	<code>protected</code>	<code>throw</code>
<code>byte</code>	<code>else</code>	<code>import</code>	<code>public</code>	<code>throws</code>
<code>case</code>	<code>enum</code> ****	<code>instanceof</code>	<code>return</code>	<code>transient</code>
<code>catch</code>	<code>extends</code>	<code>int</code>	<code>short</code>	<code>try</code>
<code>char</code>	<code>final</code>	<code>interface</code>	<code>static</code>	<code>void</code>
<code>class</code>	<code>finally</code>	<code>long</code>	<code>strictfp</code> **	<code>volatile</code>
<code>const</code> *	<code>float</code>	<code>native</code>	<code>super</code>	<code>while</code>

\* not used  
\*\* added in 1.2  
\*\*\* added in 1.4  
\*\*\*\* added in 5.0

All the chapter numbers and section headings that follow are from the following textbook:

Java 2: The Complete Reference, Fifth Edition, by Herbert Schildt

A pdf copy of this book can be found at:

\\tsr\Spring\CSE\Annajiat\Books\Java 2 The Complete Reference.pdf

Please read through all the material mentioned below before coming to lab. You will not have time to read this in lab, and might struggle with the lab tasks as a consequence.

Chapter 5, Control Statements

Java's Selection Statements, P. 100 – 104, till before “switch”