Building Java Programs

Chapter 1
Lecture 1-1: Introduction; Basic Java Programs

**reading:** 1.1 - 1.3
**self-check:** #1-14
**exercises:** #1-4
What is CSE?

- **Computer Science**
  - The study of theoretical foundations of information and computation and their implementation and application in computer systems. -- Wikipedia
  - Many subfields
    - Graphics, Computer Vision
    - Artificial Intelligence
    - Scientific Computing
    - Robotics
    - Databases, Data Mining
    - Computational Linguistics, Natural Language Processing ...

- **Computer Engineering**
  - Overlap with CS and EE; emphasizes hardware
The CS job market

SOURCES: Tabulated by National Science Foundation/Division of Science Resources Statistics; data from Department of Education/National Center for Education Statistics: Integrated Postsecondary Education Data System Completions Survey; and NSF/SRS: Sur
What is programming?

- **program**: A set of instructions to be carried out by a computer.

- **program execution**: The act of carrying out the instructions contained in a program.

- **programming language**: A systematic set of rules used to describe computations in a format that is editable by humans.
  - This textbook teaches programming in a language named Java.
Some influential ones:

- **FORTRAN**
  - science / engineering
- **COBOL**
  - business data
- **LISP**
  - logic and AI
- **BASIC**
  - a simple language
Some modern languages

- **procedural languages**: programs are a series of commands
  - **Pascal** (1970): designed for education
  - **C** (1972): low-level operating systems and device drivers

- **functional programming**: functions map inputs to outputs
  - **Lisp** (1958) / **Scheme** (1975), **ML** (1973), **Haskell** (1990)

- **object-oriented languages**: programs use interacting "objects"
  - **Smalltalk** (1980): first major object-oriented language
  - **C++** (1985): "object-oriented" improvements to C
    - successful in industry; used to build major OSes such as Windows
  - **Java** (1995): designed for embedded systems, web apps/servers
    - Runs on many platforms (Windows, Mac, Linux, cell phones...)
    - The language taught in this textbook
Basic Java programs with println statements

reading: 1.2 - 1.3
self-check: #5-14
exercises: #1-4
Compiling/running a program

1. **Write** it.
   - **code** or **source code**: The set of instructions in a program.

2. **Compile** it.
   - **compile**: Translate a program from one language to another.
   - **byte code**: The Java compiler converts your code into a format named *byte code* that runs on many computer types.

3. **Run** (execute) it.
   - **output**: The messages printed to the user by a program.
A Java program

```java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
        System.out.println();
        System.out.println("This program produces");
        System.out.println("four lines of output");
    }
}
```

- **Its output:**
  
  Hello, world!

  This program produces
  four lines of output

- **console:** Text box into which the program's output is printed.
Every executable Java program consists of a **class**, that contains a **method** named **main**, that contains the **statements** (commands) to be executed.
System.out.println

- A statement that prints a line of output on the console.
  - pronounced "print-linn"
  - sometimes called a "println statement" for short

- Two ways to use System.out.println:
  - System.out.println("text");
    Prints the given message as output.
  - System.out.println();
    Prints a blank line of output.
Names and identifiers

- You must give your program a name.

```java
public class GangstaRap {

    // Naming convention: capitalize each word (e.g. MyClass_Name)
    // Your program's file must match exactly (GangstaRap.java)
    // includes capitalization (Java is "case-sensitive")

    // identifier: A name given to an item in your program.
    // must start with a letter or _ or $
    // subsequent characters can be any of those or a number

    // legal: _myName TheCure ANSWER_IS_42 $bling$
    // illegal: me+u 49ers side-swipe Ph.D's
```
### Keywords

- **Keyword**: An identifier that you cannot use because it already has a reserved meaning in Java.

<table>
<thead>
<tr>
<th>reserved keyword</th>
<th>reserved keyword</th>
<th>reserved keyword</th>
<th>reserved keyword</th>
<th>reserved keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract</td>
<td>default</td>
<td>if</td>
<td>implements</td>
<td>private</td>
</tr>
<tr>
<td>boolean</td>
<td>do</td>
<td>import</td>
<td>protected</td>
<td>throw</td>
</tr>
<tr>
<td>break</td>
<td>double</td>
<td>instanceof</td>
<td>return</td>
<td>throws</td>
</tr>
<tr>
<td>byte</td>
<td>else</td>
<td>int</td>
<td>short</td>
<td>transient</td>
</tr>
<tr>
<td>case</td>
<td>extends</td>
<td>interface</td>
<td>static</td>
<td>try</td>
</tr>
<tr>
<td>catch</td>
<td>final</td>
<td>long</td>
<td>static</td>
<td>void</td>
</tr>
<tr>
<td>char</td>
<td>finally</td>
<td>native</td>
<td>super</td>
<td>volatile</td>
</tr>
<tr>
<td><strong>class</strong></td>
<td>float</td>
<td>new</td>
<td>switch</td>
<td>while</td>
</tr>
<tr>
<td>const</td>
<td>for</td>
<td>package</td>
<td>synchronized</td>
<td></td>
</tr>
<tr>
<td>continue</td>
<td>goto</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The reserved keywords are those that cannot be used as identifiers in Java.
Syntax

- **syntax**: The set of legal structures and commands that can be used in a particular language.
  - Every basic Java statement ends with a semicolon `;`
  - The contents of a class or method occur between `{` and `}`

- **syntax error (compiler error)**: A problem in the structure of a program that causes the compiler to fail.
  - Missing semicolon
  - Too many or too few `{` `}` braces
  - Illegal identifier for class name
  - Class and file names do not match
  ...


Syntax error example

1 public class Hello {
2     p\textcolor{red}{ooblic} static void main(String[] args) {
3         System.\textcolor{red}{owt.}println("Hello, world!")\_\_\_\_\_
4     }\n5 }

• Compiler output:

Hello.java:\textcolor{red}{2}: <identifier> expected
    p\textcolor{red}{ooblic} static void main(String[] args) {
        ^
Hello.java:\textcolor{red}{3}: ';' expected
}^\_
2 errors

• The compiler shows the line number where it found the error.
• The error messages can be tough to understand!
Strings

- **string**: A sequence of characters to be printed.
  - Starts and ends with a " quote " character.
    - The quotes do not appear in the output.
  - Examples:
    - "hello"
    - "This is a string. It's very long!"

- **Restrictions**:
  - May not span multiple lines.
    - "This is not a legal String."
  - May not contain a " character.
    - "This is not a "legal" String either."
Escape sequences

- **escape sequence**: A special sequence of characters used to represent certain special characters in a string.

  \t \t tab character
  \n \n new line character
  \\
 quotation mark character
  \\
 backslash character

- **Example:**
  ```java
  System.out.println("\\hello\\nhow\\tare "\\"you"?\\\\");
  ```

- **Output:**
  \hello
  how are "you"?\\
Questions

• What is the output of the following `println` statements?

```java
System.out.println("\ta\tb\tc");
System.out.println("\\");
System.out.println("'");
System.out.println(""");
System.out.println("C:\in\the downward spiral");
```

• Write a `println` statement to produce this output:

/ \ // \\ /// \\\
Answers

• Output of each `println` statement:

```
a    b    c
\ \  \
'  '
"""
C: in he downward spiral
```

• `println` statement to produce the line of output:

```java
System.out.println("/  \  //  \///  ///  \\
  \\
  "");
```
Questions

● What println statements will generate this output?

This program prints a quote from the Gettysburg Address.

"Four score and seven years ago, our 'fore fathers' brought forth on this continent a new nation."

● What println statements will generate this output?

A "quoted" String is 'much' better if you learn the rules of "escape sequences."

Also, "" represents an empty String. Don't forget: use \" instead of " ! " is not the same as " 
Answers

- `println` statements to generate the output:

  ```java
  System.out.println("This program prints a");
  System.out.println("quote from the Gettysburg Address.");
  System.out.println();
  System.out.println("\"Four score and seven years ago,\")
  System.out.println("our 'fore fathers' brought forth on");
  System.out.println("this continent a new nation.\"\")
  ```

- `println` statements to generate the output:

  ```java
  System.out.println("A \"quoted\" String is");
  System.out.println("'much' better if you learn");
  System.out.println("the rules of \"escape sequences.\"");
  System.out.println();
  System.out.println("Also, \"\" represents an empty String.");
  System.out.println("Don't forget: use \\\" instead of \"!");
  System.out.println("'' is not the same as \"\")
  ```