Human beings use language to communicate their thoughts, feelings, ideas and information. Languages use symbols to represent ideas. Computer software is written in a computer language that uses symbols to represent data and operations on that data. All computer programs written in a computer language share restrictions imposed on them by the nature of that language, and of languages in general.

**High-Level Languages**

Popular computer languages such as Java, C++ and Python are called *high-level languages* because their statements describe complex operations that must be broken into a sequence containing many machine instructions.

**Example**

A high-level language statement and its possible equivalent machine operations.

<table>
<thead>
<tr>
<th>High-Level Statement</th>
<th>Machine Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>if ( count &gt; 0 )</td>
<td>(1) Move the value of count from internal memory to the CPU’s logic unit</td>
</tr>
<tr>
<td>average = total / count;</td>
<td>(2) Compare it to 0</td>
</tr>
<tr>
<td></td>
<td>(3) If it’s greater than 0 jump to instruction (5)</td>
</tr>
<tr>
<td></td>
<td>(4) Jump to instruction (9)</td>
</tr>
<tr>
<td></td>
<td>(5) Move the value of total from internal memory to the CPU’s arithmetic unit</td>
</tr>
<tr>
<td></td>
<td>(6) Move the value of count to the arithmetic unit.</td>
</tr>
<tr>
<td></td>
<td>(7) Divide the values</td>
</tr>
<tr>
<td></td>
<td>(8) Move the quotient from the CPU’s arithmetic unit to the address in internal memory that holds average</td>
</tr>
<tr>
<td></td>
<td>(9) . . .</td>
</tr>
</tbody>
</table>
**Procedural Languages**

Using a *procedural language* (or *imperative language*), you must describe how to get what you want as a result. Java is a procedural language.

**Example**
The Java code to display the first and last names of all students. You need statements that loop through the student data, extracting and printing the correct items.

```java
ResultSet r = s.executeQuery( "SELECT * FROM STUDENT" );
while ( r.next( ) )
{
    System.out.print( r.getObject( 1 ) );
    System.out.println( r.getObject( 2 ) );
}
```

By contrast, a *nonprocedural language* only requires that you describe what you want, not how to get it. SQL is a nonprocedural language.

**Example**
A SQL statement to display the first and last names of all students. You need only specify that you want the first and last names from the student data.

```sql
SELECT FirstName, LastName
FROM STUDENT;
```

**Lexicon, Syntax, Semantics**

All languages, including computer languages have vocabulary, grammar and meaning. In computer science and linguistics, these concepts are referred to as *lexicon, syntax* and *semantics*, respectively.

The *lexicon* of a computer language is its total inventory of words and symbols. An item in the lexicon is called a *lexeme*, which is the basic unit of meaning in a computer program. Lexemes are made up of smaller units called *characters* that have no inherent meaning by themselves.
**Example**
This computer language phrase contains eight lexemes.  

\[
\text{if ( x \% 2 == 1 )}
\]

This one has four.  

\[
taxRate = 0.08;
\]

The lexeme `taxRate` is made up of seven characters (`t-a-x-R-a-t-e`). Each of these characters is meaningless by itself but combine together to form a variable identifier.

A *token* is a category of lexemes, rather like what we call a *part of speech* in English (e.g. noun, verb, adjective). In situations where we are interested only in the token and not the specific lexeme, we use the word *token* to refer to both.

**Example**
*Floating-point literal* is a token of which the lexeme `0.08` in the previous example is a member. Other members include `3.14159` and `2.5`.

The *syntax* of a computer language determines the correct form of its statements. Its rules dictate how to combine tokens into valid statements.

**Examples**

This is the correct way to start an *if* statement in Java:

\[
\text{if ( x \% 2 == 1 )}
\]

The syntax rules of Java say that the *if* lexeme must be followed by a parenthesized expression. So the Java compiler wouldn’t understand this:

\[
\text{if x \% 2 == 1}
\]

The *semantics* of a language is the meaning of its statements, i.e. the operations specified by the statements.
Program Compilation

A compiler is a software tool that translates statements of a high-level language into equivalent machine instructions.

```java
if ( count > 0 )
    average = total / count;
```

(1) Move the value of count from internal memory to the CPU
(2) Compare it to 0
(3) If it's greater than 0 jump to instruction (5)
    . . .

It works through several phases, inputting the high-level program character by character, determining its meaning and generating the equivalent machine code. These phases are shown in the picture on the next page.

The first phase, called lexical analysis (or scanning), involves reading the individual characters of the computer program, building each lexeme and identifying the token to which it belongs.

The second phase is called syntax analysis (or parsing) and involves combining tokens into a recognizable program statement.

In the third phase the compiler translates the program statement into the equivalent machine code instructions, a process called code generation.
The formation of tokens and statements in a computer language is guided by very precise rules. Small typographical errors in a program can result in the compiler rejecting it. Attention to detail is important in computer programming.