Example 8: Scanning Names

Implement the method readName(t) according to its specification in Chapter 2. This method is called by next() to recognize a name or keyword. The token kind should be stored in t.kind and the text of the name in t.val.

Solution

When readName() is called, ch contains the first character of a name (i.e., a letter). readName() must now read further characters, digits and underscores and build a name from them, which is then stored in t.val.

Names and keywords have the same lexical structure. After a name has been recognized, we must therefore check whether it is a keyword. We use binary search for this, although you could also use a hash table. To do this, we declare a sorted array key with the MicroJava keywords and, in parallel, an array keyVal with the corresponding token codes:

```
private static final String key[] =
    { "break", "class", "else", "final", "if", "new", "print", "program", "read", "return", "void", "while" };
private static final int keyVal[] =
    { break_, class_, else_, final_, if_, new_, print_, program_, read_, return_, void_, while_ };
```

We also declare a char array lex in which we collect the characters of the name:

```
private static char[] lex = new char[128];
```

The method readName() then looks as follows:

```
private static void readName (Token t) {
    int i = 0;
    do {
        lex[i++] = ch; nextCh();
    } while ('a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' || '0' <= ch && ch <= '9' || ch == '_');
    t.val = new String(lex, 0, i);
    t.kind = checkIfKeyword(t.val);
}
```

The method checkIfKeyword() looks up the name in a keyword table key using binary search. If the name is found, the corresponding keyword code is returned, otherwise ident.

```
private static int checkIfKeyword (String s) {
    int i = 0, j = key.length-1;
    while (i <= j) {
        int k = (i + j) / 2;
        int d = s.compareTo(key[k]);
        if (d < 0) j = k - 1;
        else if (d > 0) i = k + 1;
        else return keyVal[k];
    }
    return ident;
}
```