Names, Parsing, Scanning

Professor Don Colton

Brigham Young University Hawaii

This seems like a good time to tell why we have such strict rules on variable names. (This section is not really about data. It's about naming data and why there are issues with it.)

As you will recall, a variable name must start with a letter. (Letters include A through Z, a through z, and underscore.) It can continue with letters and digits. (Digits include 0 through 9.) It has a dollar sign as a prefix. It must not include special characters or spaces.

Why is that?

1 Computers are Stupid

In section 1.1 we established that computers are stupid. Humans understand things by looking for meaning. This allows us to overlook mistakes that others might make.

For example, imagine a traffic light, red on top, yellow in the middle, green on the bottom. Now imagine the red glass lens covering the top light is broken and missing. When the light should be red it is white. You drive up to the light. The white light (on top) is shining. What should you do?

Normally the human would conclude the light was supposed to be treated as though it were red. The human looks for the meaning and overlooks what might be seen as trivial variances.

Computers generally fail to pass that kind of a test. Computers generally are programmed to look for specific conditions and respond to them. Slightly different conditions can result in failure to properly recognize the situation.

2 Parsing

Vocabulary warning: this section introduces lots of new words that are only used in this section.

Computer programs are typically written in text. They are then converted into a runnable program through a process called compiling. Compiling is done by a compiler.

Parsing is the activity your compiler does when it reads in your program and tries to understand it. The first step in parsing is called lexical scanning. In this step, the text of your computer program is divided up into individual words, called tokens.

In the next step, the tokens are organized according to the grammar of your programming language.

Let's look at an example.

```
print "Hello, World!";
```

In this example, the lexical scanner would break up the program into tokens. It would find the following tokens:

```
print
"Hello, World!"
;
```

Tokens are identified by the spaces between them and by the punctuation around them. The same tokens would be found if the program looked like this:

```
print"Hello, World!"   ;
```

Specifically, the amount of white space between tokens is not significant. If you can have one white space, you can have lots of white space. It's all the same to the compiler.

3 Variable Names

For the parser to work efficiently, variable names and other tokens must follow simple rules. The simple rule is that viable names are prefixed with a dollar sign, start with a letter (including underscore), and continue with letters and digits.