Input: Hello, Joe!

Professor Don Colton

Brigham Young University Hawaii

Our task is to make the computer say hello, calling the user by name.

Here is the first version of the program.

```
print "What is your name?";
$name = <STDIN>;
print "Hello, $name!";
$wait = <STDIN>;
```

We are familiar with most of this. The new part is the `"Hello, $name!"` part.

Inside double quotes the dollar sign introduces the variable `$name` which will contain whatever the user typed on the previous line.

Type it in and run it.

Notice that when you run it, if you type in `Joe`, the output will be like this:

```
What is your name?
Joe
Hello, Joe
!
```

Strangely the exclamation mark is on a different line. Why is that?

It turns out that the input, `<STDIN>`, returns everything that was typed, including the `enter` on the end of the line. When `$name` is printed as part of the literal, we get J, o, e, enter, in place of the `$name`.

We need to get rid of that pesky enter at the end of the line.

Fortunately Perl has a command for getting rid of enters at the end of lines. It is called `chomp`. (Yeah, I think it sounds a little silly. These things happen.)

Here is our new program:

```
print "What is your name?";
$name = <STDIN>;
chomp ( $name );
print "Hello, $name!";
$wait = <STDIN>;
```

Here is the new result:

```
What is your name?
Joe
Hello, Joe!
```

What would happen if we try this?

```
print "What is your name?";
$name = <STDIN>;
chomp ( $name );
chomp ( $name );
chomp ( $name );
print "Hello, $name!";
$wait = <STDIN>;
```

Chomp only removes the enter (also called a carriage return) if it is present. After the first chomp, the other chomps have no additional effect. The program will run the same as with one chomp.