Shell Expansion (1)

Shell variables

In the shell, `VARIABLE=VALUE` sets a variable to a value and `$VARIABLE` retrieves its value. For example, to save typing a filename twice:

```
p=arguments
gcc -Wall $p.c -o $p
```

which expands to `gcc -Wall arguments.c -o arguments`. If you want to use a variable inside a word, you can use curly braces: `${a}b` means the value of the variable `a` followed by the letter b, whereas `$ab` would mean the value of the variable `ab`.

It is good practice to double-quote variables used like this, because if you tried for example to compile a program called `silly name.c` with a space in its name, then

```
program="silly name"
gcc -Wall $program.c -o $program
```

would expand to

```
gcc -Wall silly name.c -o silly name
```

and this would confuse your compiler because you are telling it to compile three source files called `silly`, `name.c` and `name` to a program called `silly`. Correct would be:

```
program="silly name"
gcc -Wall "$program.c" -o "$program"
```

which expands to

```
gcc -Wall "silly name.c" -o "silly name"
```

which does what you want - if you indeed want a program with a space in its name!

There is no harm in double-quoting a shell variable every time you want to use it, and this is good practice as it still works if the variable is set to a value that contains spaces.

Note that we also had to quote setting the variable name in the first place, because

```
program=silly name
```

would translate as: set the variable `program` to the value `silly`, then execute the program `name`. Variable assignments only apply to the first argument following them, although you can assign more than one variable.

Note that this does not work as expected either:

```
file=arguments gcc -Wall "$file.c" -o "$file"
```

The problem here is that the shell first reads the line and substitutes in the value of `$file` (unset variables expand to the empty string by default) before starting to execute the command, so you are reading the variable's value before writing it. Leaving off the quotes doesn't help: you need to set the variable on a separate line.