Introduction to Software Systems (CS6.201)

Summer 2021, IIIT Hyderabad

01 June, Tuesday (Lecture 3)

Taught by Sai Anirudh Karri

Shell Variables

Variables can be declared using the `declare v_name` syntax. They have no datatype. Assignment is done using the `=` operator.
Variables are lost as soon as the terminal window is closed. If they need to be permanent, they need to be added to the `.bashrc` file.
To print or access the value of a variable, its name must be prefixed with `$`.

If statements have the following syntax:

```
if test [-flags] <expression>
    then
    block
fi
```

The `test` command evaluates the expression. It returns 0 if the expression is true, 1 if it is false or missing, and > 1 if an error occurs.
When a variable is passed, it must be enclosed in double quotes, as in `"$v_name"`.

There are some special variables; for example,

- `$0` holds the name of the command being executed and `$1` to `$9` hold the command-line arguments passed to the executable file.
- `$#` holds the number of arguments passed.
- `$@` holds the list of arguments passed.
- `$?` holds the error code of the previous command; it is 0 in case the command was executed successfully.
- `$$` holds the PID of the command.

Piping and Redirection

We can access the output of a command *as a variable* using the `$(command)` syntax, as in `$(pwd)` or `$(date)`.
We use piping to pass the output of one command to another command as input. For example, if we want to know the number of files in a folder, we can run `wc` to count the number of words or lines in the output of the `ls` command, as `ls | wc` or `ls -a | wc -l`. We can also redirect output to a file, as in `cat hello.txt > /dev/stdout`. `>` redirects the output of the first command; we can also use `2>` to redirect its error code.
To redirect an environmental variable to a file, we need to use `>>`; for example, `echo $HOSTNAME >> $HOSTNAME"_stats.txt"`.

To run a command *inside* an `echo` statement, use backticks, as in `echo \`uname -a\` > file.txt`.

Maths Expressions

There are multiple ways to evaluate mathematical expressions in the shell.
1. `let v_name=<expression>`. This does not print the value; to print it, we run `echo $v_name`.
2. `expr <expression>` evaluates and prints the value, but the values in the expression should be space-separated (they are distinct arguments to `expr`).
3. `(( <expression> ))` results in the evaluation of the expression as well.
4. The basic calculator `bc`, for example `echo "10+5" | bc`. Expressions have to be piped to it.