```
var1="spreadsheet.csv"
```

↳ the variable contains the file <u>name</u> (string of 15 characters)

```
var2=$(cat spreadsheet.csv)
```

↳ this variable contains the whole <u>content</u> of the file (very large string containing some \n chars)

Concatenation in Bash: the no-operator

```
line1="Hello, Margaret"
line2="Thatcher"

line3=${line1}${line2}
echo "${line3}"
```

→ produces "Hello, MargaretThatcher"

REGULAR EXPRESSIONS

Regular expressions help specify strings loosely, according to certain patterns.

REGEXP	WHAT IT CAN MATCH
a	the character "a" only
[abd]	any one of the single characters in the brackets
[^abd]	any single character **NOT** being a, b or d

REGEXP	WHAT IT MATCHES
.	any single character
a*	any series of zero or more consecutive 'a' characters
[0-9]	any single digit
[0-9]*	any number of digit(s) (zero or more)
[0-9]+	one or more digit(s)
.*	any string, whichever its length

⚠ extended regexps

Structure of an AWK script

BEGIN { actions to be performed once at the beginning } (optional bits in green)

selection rule1 { actions1 }

selection rule2 { actions 2 }

selection rule3 { actions3 }

END { actions to be performed after the last record is processed }

```
BEGIN { FS=","; sum=0 }
{ sum += $2 }
END { print sum }
```

→ this script prints the sum of all values in the second field