**Special Characters**

What makes a character *special*? If it has a meaning beyond its *literal meaning*, a meta-meaning, then we refer to it as a *special character*

Comment

**#** Lines beginning with a # (with the exception of #!) are comments and will *not* be executed & extra #! Act as a comment.

1 # This line is a comment.

Command separator **;**

Permits putting two or more commands on the same line

Terminator in **Case** option **;;**

Curent directory **.** Parent directory **..**

Double Quotes

**“$HOME“** Prevent all **special** character displays ==> /usr/home

Single quotes

**‘$HOME’**: skips special chara display ==> $HOME

Coma Operator **,**

comma links together a series of arithmetic operations. All are evaluated, but only the last one is returned. The *comma* operator can also concatenate strings.

**Lower case conversion** **, ,,**

var=”SPORTZ”

echo ${var,} >>> first car lowercase >>> sPORTZ

echo ${var,,} >>> All lower case >>> sportz

**Upper Case operation** ^ ^^

Var=”sportz”

echo ${var^} >>> first car lowercase >>> Sportz

echo ${var^^} >>> All lower case >>> SPORTZ

**Colon :**

This is the shell equivalent of a "NOP" (no op, a do-nothing operation).

The "**:**" command is itself a Bash builtin, and

1 :

2 echo $? # 0 its exit status is true (0).

**A colon can serve as a placeholder in an otherwise empty function**.

1 not\_empty ()

2 {

3 :

4 } # Contains a : (null command), and so is not empty.

Redirection > **>> &>** redirects both stout and sterr

Input Redirection << <<<

Force Redirection even clob is set **>|**

**Backslash**

\ escapes the character

A quoting mechanism for single characters

**Forward slash**

/ Filename path separator

Separates the components of a filename (as in /home/bozo/projects/Makefile).

Reverse **!**

**Pipe |**

Test operator **?**

Dollar **$**

**Represents command substitution $(command)**

**Expand and evaluate integer expression between (( ))**

Represents Varaible/Parameter substitution **${variable name}**

In a regular expression, a "$" addresses the end of a line of text.

Positional parameter **$\*,$@**

Exit status  **$?**

Process ID variable **$$**

Command group ()

Block of codes {}

Test expression [ ] and [[ ]]

Array element [ ]

Run job in background &

AND logical operator &&

BODMUS

EOF (end-of-file). This also terminates input from stdin