BASH

SECTION 2

VARIABLES AND SHELL EXPANSIONS

SECTION CHEAT SHEET

PARAMETERS

DEFINITION:
Parameters are entities that store values

THERE ARE 3 TYPES OF PARAMETERS

- VARIABLES
- POSITIONAL PARAMETERS
- SPECIAL PARAMETERS

VARIABLES

DEFINITION:
Variables are parameters that you can change the value of

2 TYPES OF VARIABLES

1. USER-DEFINED VARIABLES
2. SHELL VARIABLES
   - BOURNE SHELL VARIABLES
   - BASH SHELL VARIABLES

SETTING THE VALUE OF A VARIABLE

```
name=value
```

Note 1: There should be no spaces around the equals sign
Note 2: Names of user-defined variables should be all lowercase

SOME COMMON SHELL VARIABLES

HOME	Absolute path to the current user's home directory
PATH	List of directories that the shell should search for executable files
USER	The current user's username
HOSTNAME	The name of the current machine
HOSTTYPE	The current machine's CPU architecture
PS1	The terminal prompt string

Link to a list of Bourne shell variables

Link to a list of Bash shell variables

PARAMETER EXPANSION

SYNTAX:

- Simple Syntax: `$parameter`
- Advanced Syntax: `${parameter}`

DEFINITION:

Parameter expansion is used to retrieve the value stored in a parameter

PARAMETER EXPANSION TRICKS

1. `${parameter^}`
   Convert the first character of the parameter to uppercase
2. `${parameter^^}`
   Convert all characters of the parameter to uppercase
3. `${parameter,}`
   Convert the first character of the parameter to lowercase
4. `${parameter,,}`
   Convert all characters of the parameter to lowercase
5. `${#parameter}`
   Display how many characters the variable’s value contains
6. `${parameter : offset : length}`
   The shell will expand the value of the parameter starting at character number defined by “offset” and expand up to a length of “length”

Note: None of these alter the value stored in the parameter. They just change how it is displayed after the expansion.

Link to list of more parameter expansion tricks

COMMAND SUBSTITUTION

DEFINITION:
Command Substitution is used to directly reference the result of a command

Syntax for command substitution

```
$(command)
```

ARITHMETIC EXPANSION

DEFINITION :
Arithmetic Expansion is used to perform mathematical calculations in your scripts.

Syntax for Arithmetic Expansion

```
$(( expression ))
```

ARITHMETIC OPERATORS RANKED IN ORDER OF PRECEDENCE (HIGHEST PRECEDENCE FIRST):

OPERATOR(S)	MEANING(S)	COMMENTS
( )	Parentheses	Anything placed in parentheses is given the highest precedence and is always run first.
**	Exponentiation. 2**4 means 2 to the power of 4, which is 16	
*, /, and %	Multiplication, Division, and Modulo. Modulo calculates the remainder of a division.	These have the same precedence.
+ and -	Addition and substraction	These have the same precedence.

Note: When two operators have the same precedence, the one furthest to the left gets performed first.

THE BC COMMAND

Using the bc command

```
echo “expression” | bc
```

Using the scale variable to control the number decimal places shown

```
echo “scale=value; expression” | bc
```

TILDE EXPANSION

DEFINITION:
Tilde expansion provides various shortcut for referencing folders on the command line.

SYNTAX	MEANING
~	The current value of the **$HOME** shell variable (usually the current user's home directory)
~username	If username refers to a valid user, give the path to that user's home directory
~-	The current value stored in the **$OLDPWD** shell variable
~+	The current value stored in the **$PWD** shell variable

Note: **$PWD** stores the current working directory and **$OLDPWD** stores the previous working directory

BRACE EXPANSION

DEFINITION:
A way of automatically generating text according to a certain pattern.

SYNTAX	MEANING
{1,2,3,4,5}	1 2 3 4 5
{1..5}	1 2 3 4 5
{a..e}	a b c d e
{1..5..2}	1 3 5
Month{01..12}	Month01, Month02, Month03, Month04, Month05, Month06, Month07, Month08, Month09, Month10, Month11, Month12
file{1..5}.txt	file1.txt file2.txt file3.txt file4.txt file5.txt
~/{Documents,Downloads}/file{1..2}.txt	~/Documents/file1.txt ~/Documents/file2.txt ~/Downloads/file1.txt ~/Downloads/file2.txt

Note: There should be no spaces around any commas or double dots (..)