Program Structure

Making a file a bash script

Put the following line at beginning of file.

```
#!/bin/bash
```

Comments

```
# This is a bash comment
```

Functions

```
function function_name {
  function_body
}
```

Alternate

```
function_name() {
  function_body
}
```

Using arguments

```
function_name() {
  echo "$# is the number of arguments received";
  echo "$1 $2 $3 ... are the first, second, third... arguments";;
}
```

Data Types

Variables in bash are not associated with any data type. All data is essentially string of characters. However, at evaluation time, depending on the context (operator), the strings are interpreted as either characters ('a', '5',...), strings ("xyz", "pqr",...), integers (45, -348,...) or floating point expressions (3.45, 0.009...).

Operators

Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Remainder
++	pre/post increment

--	pre/post decrement

Relational Operators

On numbers

Operator	Description
-lt	Less Than
-le	Less Than or Equal To
-gt	Greater Than
-ge	Greater Than or Equal To
-eq	Equal To
-ne	Not Equal To

On strings

(String comparisons are in ASCII order)

Operator	Description
<	Less Than
<=	Less Than or Equal To
>	Greater Than
>=	Greater Than or Equal To
==	Equal To
!=	Not Equal To
-z	True if string length is zero
-n	True if string length is non-zero

Logical Operators

Operator	Description
&	Bitwise AND
^	Bitwise XOR
|	Bitwise OR
&&	Logical AND
||	Logical OR

Flow of Control

Conditionals

If statements

```
if [[ expression_1 ]]; then
   statements_1
elif [[ expression_2 ]]; then
   statements_2
...
...
else
```

```
   statements_n
fi
```

Case statements

```
case expression in
case1) statements1;;
case2) statements2;;
....
caseN) statementsN;;
esac
```

Loops

For loop

```
for i in `seq 1 10`;
do
   echo $i
done
```

While loop

```
while [ condition ]; do
   statements;
done
```

Until loop

```
until [ condition ]; do
   statements;
done
```

Operations

Language Specifics

Expression Evaluation

Arithmetic

((expression))

```
A=$(( 10 * (2 + 2) ))
# A now holds the value 40
```

Conditional

[[expression]]

```
if [[ $age -gt 100 ]]; then
  echo "100 years completed!"
fi
```

Conditional checks on files

Common checks

Operator	True if
-a / -e	File exists
-d	File exists and is a directory
-f	File exists and is a regular file
-s	File exists and has size greater than zero
-r	File exists and is readable
-w	File exists and is writable
-x	File exists and is executable
-N	File exists and has been modified since it was last read

File type related checks

Operator	True if
-h/ -L	File exists and is a symbolic link
-b	File exists and is a block special file
-c	File exists and is a character special file
-p	File exists and is a named pipe (FIFO)
-S	File exists and is a socket

Checks on other file properties

Operator	True if
-O	File exists and is owned by the effective user id
-G	File exists and is owned by the effective group id
-g	File exists and is set-group-id
-k	File exists and its sticky bit is set