Experience certainty

Confidentiality Statement

This document contains confidential information of Tata Consultancy Services Limited, which is provided for the sole purpose of permitting the recipient to evaluate the proposal submitted herewith. In consideration of receipt of this document, the recipient agrees to maintain such information in confidence and to not reproduce or otherwise disclose this information to any person outside the group directly responsible for evaluation of its contents, except that there is no obligation to maintain the confidentiality of any information which was known to the recipient prior to receipt of such information from Tata Consultancy Services Limited, or becomes publicly known through no fault of recipient, or is received without obligation of confidentiality from a third party owing no obligation of confidentiality to Tata Consultancy Services Limited.

Experience certainty.



Tata Code of Conduct

We, in our dealings, are self-regulated by a Code of Conduct as enshrined in the Tata Code of Conduct. We request your support in helping us adhere to the Code in letter and spirit. We request that any violation or potential violation of the Code by any person be promptly brought to the notice of the Local Ethics Counselor or the Principal Ethics Counselor or the CEO of TCS. All communication received in this regard will be treated and kept as confidential.



TATA CONSULTANCY SERVICES



Virtual ILP - Shell Programming

Content Manual

Version 1.1

December 2014

(ILP Guwahati)

Experience certainty

Table of Content

1.4 Basic Operators in shell scripting	5
1.4.1 Arithmetic Operators	
1.4.2 Relational Operators	
1.4.3 Boolean Operators	
1.4.4 String Operators	
1.4.5 File Test Operators:	

TCS Internal 4

Experience certainty

1.4 Basic Operators in shell scripting

Below operators are supported by shell.

- Arithmetic Operators.
- Relational Operators.
- Boolean Operators.
- String Operators.
- File Test Operators.

Here is simple example to add two numbers:

Example:

#!/bin/sh

 $val=\exp 2 + 2$

\$ echo "Total value : \$val"

Output:

\$ Total value: 4

There are following points to note down:

- There must be spaces between operators and expressions for example 2+2 is not correct, where as it should be written as 2+2.
- Complete expression should be enclosed between ``, called inverted commas to execute expr command correctly.

1.4.1 Arithmetic Operators

Assume variable a holds 10 and variable b holds 20 then:

Operator Description	Example
----------------------	---------

TCS Internal

TATA CONSULTANCY SERVICES Experience certainty.

+	Addition	`expr \$a + \$b ` will give 30
-	Substraction	`expr \$a - \$b ` will give -10
*	Multiplication	`expr \$a * \$b` will give 200
/	Division	`expr \$b / \$a` will give 2
%	Modulus	`expr \$a % \$b` will give 0
!=	Not equal	[\$a != \$b] will give true
=	assignment	a=\$b will assign value of b to a.
==	Equality	[\$a == \$b] will return false.

It is very important to note here that all the conditional expressions would be put inside square braces with one spaces around them, for example [\$a == \$b] is correct where as [\$a == \$b] is incorrect.

1.4.2 Relational Operators

Below are relational operators which are specific to numeric values. These operators would not work for string values unless their value is numeric.

For example, following operators would work to check a relation between 10 and 20 as well as in between "10" and "20" but not in between "ten" and "twenty".

Assume variable a holds 10 and variable b holds 20 then:

Operator	Description	Example
-eq	Check if the values of 2 operands are equal or not, if yes then condition becomes true.	[\$a -eq \$b] is false
-ne	Check if the values of 2 operands are equal or not, if values are not equal then condition becomes true.	[\$a -eq \$b] is true
-gt	Check if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	[\$a -gt \$b] is false
-lt	Check if the value of left operand is less than the value of right operand, if yes then condition becomes true.	[\$a -lt \$b] is true
-ge	Check if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	[\$a -ge \$b] is false

TATA CONSULTANCY SERVICES Experience certainty

-le	Check if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	[\$a -le \$b] is true	

It is very important to note here that all the conditional expressions would be put inside square braces with one spaces around them, for example [$\$a \le \b] is correct where as [$\$a \le \b] is incorrect.

1.4.3 Boolean Operators

Assume variable a holds 10 and variable b holds 20 then

Operator	Description	Example
!	This is logical negation. This inverts a true condition into false and vice versa.	[!false] is true
-0	This is logical OR. If one of the operands is true then condition would be true.	[\$a -lt 20 -o \$b -gt 100] is true
-a	This is logical AND. If both the operands are true then condition would be true otherwise it would be false.	[\$a -lt 20 -a \$b -gt 100] is false.

1.4.4 String Operators

These are string operators. Assume variable a holds "abc" and variable b holds "efg" then:

Operator	Description	Example
-Z	Check if the given string operand size is zero. If it is zero length then it returns true.	[-z \$a] will return false.
-n	Check if the given string operand size is non-zero. If it is non-zero length then it returns true.	[-z \$a] will return true.
=	Check if the value of two operands is equal or not, if yes then condition becomes true.	[\$a = \$b] will return false
!=	Check if the value of two operands is equal or not, if the values are not equal then condition becomes true.	[\$a != \$b] will return true

TCS Internal

str	Check if the str is not the empty string. If it is empty	[\$a] will return true
	then it returns false.	

1.4.5 File Test Operators:

Assume a variable file holds an existing file name "test" whose size is 100 bytes and has read, write and execute permission on:

Operator	Description	Example
-b file	Returns true, if file is a block special file	[-b \$file] is false.
-c file	Returns true, if file is a character special file	[-b \$file] is false.
-d file	Returns true, Check if file is a directory	[-d \$file] is not true.
-f file	Returns true, Check if file is an ordinary file or special file	[-f \$file] is true.
-r file	Returns true, Checks if file is readable	[-r \$file] is true.
-w file	Returns true, Check if file is writable	[-w \$file] is true.
-x file	Returns true, Check if file is execute	[-x \$file] is true.
-s file	Returns true, Check if file has size greater than 0	[-s \$file] is true.
-e file	Returns true, Check if file exists	[-e \$file] is true.