LUA - STRINGS

String is a sequence of characters as well as control characters like form feed. String can be initialized with three forms which includes −

- Characters between single quotes
- Characters between double quotes
- Characters between [[and]]

An example for the above three forms are shown below.

```
string1 = "Lua"
print("\"String 1 is\"",string1)

string2 = 'Tutorial'
print("String 2 is",string2)

string3 = [["Lua Tutorial"]]
print("String 3 is",string3)
```

When we run the above program, we will get the following output.

```
"String 1" is Lua
String 2 is Tutorial
String 3 is "Lua Tutorial"
```

Escape sequence characters are used in string to change the normal interpretation of characters. For example, to print double inverted commas "" , we have used \" in the above example. The escape sequence and its use is listed below in the table.

Escape Sequence	Use
\a	Bell
\b	Backspace
\f	Formfeed
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\\	Backslash
\"	Double quotes
\'	Single quotes
\[	Left square bracket
\]	Right square bracket

String Manipulation

Lua supports string to manipulate strings −

S.N.	Method & Purpose
1	**string.upper***argument* Returns a capitalized representation of the argument.
2	**string.lower***argument* Returns a lower case representation of the argument.
3	**string.gsub***mainString, findString, replaceString* Returns a string by replacing occurrences of findString with replaceString.
4	**string.strfind***mainString, findString, optionalStartIndex, optionalEndIndex* Returns the start index and end index of the findString in the main string and nil if not found.
5	**string.reverse***arg* Returns a string by reversing the characters of the passed string.
6	**string.format**... Returns a formatted string.
7	**string.char***arg* **and string.byte***arg* Returns internal numeric and character representations of input argument.
8	**string.len***arg* Returns a length of the passed string.
9	**string.rep***string, n***)** Returns a string by repeating the same string n number times.
10	**..** Thus operator concatenates two strings.

Now, let's dive into a few examples to exactly see how these string manipulation functions behave.

Case Manipulation

A sample code for manipulating the strings to upper and lower case is given below.

```
string1 = "Lua";

print(string.upper(string1))
print(string.lower(string1))
```

When we run the above program, we will get the following output.

```
LUA
lua
```

Replacing a Substring

A sample code for replacing occurrences of one string with another is given below.

```
string = "Lua Tutorial"

-- replacing strings
newstring = string.gsub(string,"Tutorial","Language")
print("The new string is",newstring)
```

When we run the above program, we will get the following output.

```
The new string is Lua Language
```

Finding and Reversing

A sample code for finding the index of substring and reversing string is given below.

```
string = "Lua Tutorial"

-- replacing strings
print(string.find(string,"Tutorial"))
reversedString = string.reverse(string)
print("The new string is",reversedString)
```

When we run the above program, we will get the following output.

```
5 12
The new string is lairotuT auL
```

Formatting Strings

Many times in our programming, we may need to print strings in a formatted way. You can use the string.format function to format the output as shown below.

```
string1 = "Lua"
string2 = "Tutorial"

number1 = 10
number2 = 20

-- Basic string formatting
print(string.format("Basic formatting %s %s",string1,string2))

-- Date formatting
date = 2; month = 1; year = 2014
print(string.format("Date formatting %02d/%02d/%03d", date, month, year))

-- Decimal formatting
print(string.format("%.4f",1/3))
```

When we run the above program, we will get the following output.

```
Basic formatting Lua Tutorial
Date formatting 02/01/2014
0.3333
```

Character and Byte Representations

A sample code for character and byte representation, which is used for converting the string from string to internal representation and vice versa.

```
-- Byte conversion

-- First character
print(string.byte("Lua"))

-- Third character
print(string.byte("Lua",3))

-- first character from last
print(string.byte("Lua",-1))

-- Second character
print(string.byte("Lua",2))

-- Second character from last
print(string.byte("Lua",-2))

-- Internal Numeric ASCII Conversion
print(string.char(97))
```

When we run the above program, we will get the following output.

```
76
97
97
117
117
a
```

Other Common Functions

The common string manipulations include string concatenation, finding length of string and at times repeating the same string multiple times. The example for these operations is given below.

```
string1 = "Lua"
string2 = "Tutorial"

-- String Concatenations using ..
print("Concatenated string",string1..string2)

-- Length of string
print("Length of string1 is ",string.len(string1))

-- Repeating strings
repeatedString = string.rep(string1,3)
print(repeatedString)
```

When we run the above program, we will get the following output.

```
Concatenated string LuaTutorial
Length of string1 is  3
LuaLuaLua
```