Module	CS4012 – Representation and Modelling
Day	3
Lab	3
Topic	Operators

Summary

(Operator): Special symbols that perform specific operations on one, two, or three values (or *operands*), and produce a single result.

Certain operators are used far more frequently than others; namely, Arithmetic, Relational and Logical operators. Perhaps the most common is the Assignment operator, which you have encountered previously. When operators appear on the same line, the order in which they are evaluated depends on their precedence. With this in mind, the following focuses on the most frequently encountered operators, and on the precedence of their evaluation when combined.

Exercise 1: evaluating Arithmetic operators

Lua supports the usual arithmetic operators: the binary '**+**' (addition), '**-**' (subtraction), '*****' (multiplication), '**/**' (division), '**%**' (modulo or remainder) and '**^**' (exponentiation).

Tasks:

1. Write a program that performs the operations listed above (on two variables) and prints the result (assigned to a third variable) to the screen.

The following code can be used as a template:

```
a = 21
b = 10
c = 0

c = a + b
print("Line 1 - Value of c is " ..  c)            - - > Line 1 – Value is 31
```

2. Evaluate the following expression:

$$1 / 4 * 2 \wedge 2 + 3$$

a. Add two pairs of parentheses that leave the value of the expression unchanged.
b. Add one pair of parentheses so that the expression evaluates to double the original value.

Tip: Use pen and paper – code it up after to check your answers.

3. In the 12-hour clock, the day is divided into two 12-hour periods. If the time now is 7pm, find the time in 6 hours. Write a program that performs the necessary operation and prints the result.

Hint: There is no 13 o'clock in the 12-hour clock (7 + 6 = 13), the time wraps around every 12 hours. The time is what remains after division by 12 - or time modulo 12.

Exercise 2: evaluating Relational operators

Lua provides the following relational operators: the binary '**<**' (less than), '**>**' (greater than), '**<=**' (less than or equal to), '**>=**' (greater than or equal to), '**==**' (equal to) and '**~=**' (not equal to)

All of these operators result in Boolean **true** or **false** values.

Task:

Write a program that evaluates the expressions below and prints the results to the screen. Try to predict the outcome of each one before running your program. Test each one separately, by commenting-out the others – some result in errors.

a. "0"== 0
b. 2 < 15
c. "2" < "15"
d. 2 <" 15"
e. 2 <= 1
f. 3 >= 3
g. true == 1
h. true >= 1
i. false == false

The output to (a) should look something like the following: *"0" == 0 is value*.

Hint: A line-comment starts with a double hyphen (--) and runs until the end of the line. The corresponding block comment starts with --[[and run until the matching]].

Exercise 3: evaluating Logical operators

Lua provides the following logical operators: the binary '**and'**, '**or**', and the unary '**not**'.

- All logical operators consider **false** and **nil** as false, and any other value as true.
- The "**and"** operator returns its first argument if it is false; if not, it returns its second argument.
- The "**or"** operator returns its first argument if it is true; if not, it returns its second argument.

Example: "true **and** 2" can be read 'true' *logical and* 'true'. The first argument is true therefore return the second argument, "2".

Task:

Evaluate the following expression:

(x > y) **and** x **or** y

a. If x = 3, y = 7
b. If x = 7, y = 3
c. What can be concluded about the above expression?

Hint: The expression is equivalent to (((x **>** y) **and** x) **or** y). In terms of precedence, not > and > or.