

## Programming #2 -- Short Circuit Evaluation

In this programming assignment the task was to check whether different languages used short circuit evaluation. This was determined by checking AND clauses in Boolean situations. You can see whether a language is short circuit by checking if the language stops evaluating after the first false clause. The different languages checked were ADA, C Shell, PHP, and Perl. All languages tested were proved to be short circuited. ADA's and then clause is short circuited, however, ADA's and clause is not short circuited.

Language	Is the language short circuit?
ADA	Yes
C Shell	Yes
PHP	Yes
Perl	Yes

ADA Test:

```
-- Xiana Lara
-- September 9th, 2020
-- Programming 2: Short Circuit Evaluation for ADA

-- This program tests ADA to see if the language implementation
-- has short circuit evaluation in the AND Boolean construct.
-- If ADA is short circuited it evaluates only the first function
-- to determine if the expression is true or false.

-- Input:  None
-- Output: 2 different and expressions
-- Expression 1: False (False && True)
-- Expression 2: False (True && False)

-- for use on CS machines
With Gnat.IO;
use Gnat.IO;

procedure adatest is

  function func1 return Boolean is
    x : Boolean;
```

```
begin

    x := False;
    -- print statement
    Put_Line("func1 has been evaluated");
    return x;

end func1;
-- of func1

function func2 return Boolean is
    y : Boolean;
begin

    y := True;
    -- print statement
    Put_Line("func2 has been evaluated");
    return y;

end func2;
-- of func2

-- main: depending on indention
begin

    -- print statement
    Put_Line("Statement: if func1 and then func2 then");

    -- result: false
    if func1 and then func2 then
        -- print statement
        Put_Line("True");
    else
        -- print statement
        Put_Line("False");
    end if;

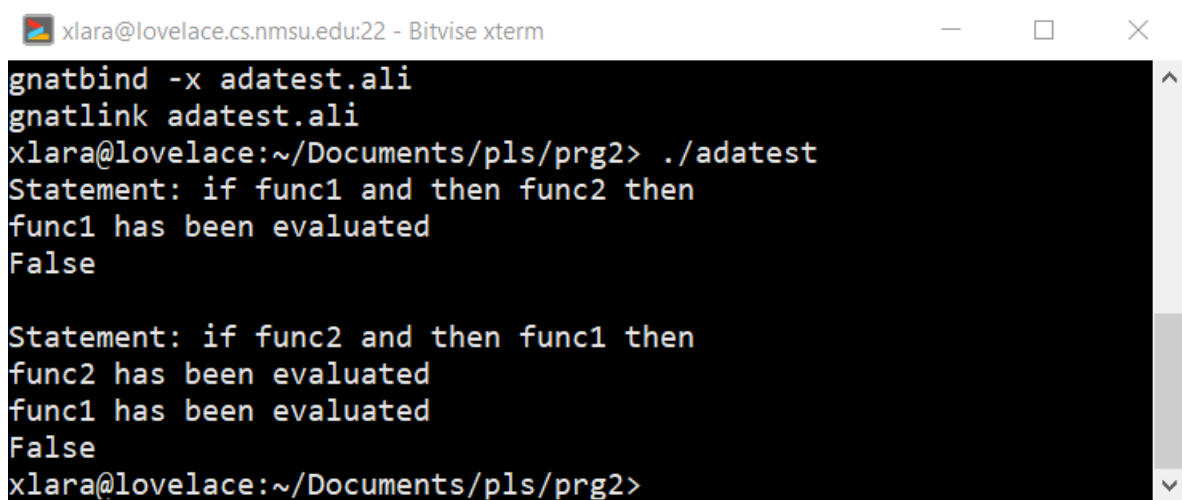
    New_Line;
    -- print statement
    Put_Line("Statement: if func2 and then func1 then");

    -- result: false
    if func2 and then func1 then
```

```
-- print statement
Put_Line("True");
else
  -- print statement
  Put_Line("False");
end if;

end adatest;
-- of main test
```

ADA Test Results:



The screenshot shows a terminal window titled "xlara@lovelace.cs.nmsu.edu:22 - Bitwise xterm". The terminal displays the following commands and output:

```
gnatbind -x adatest.ali
gnatlink adatest.ali
xlara@lovelace:~/Documents/pls/prg2> ./adatest
Statement: if func1 and then func2 then
func1 has been evaluated
False

Statement: if func2 and then func1 then
func2 has been evaluated
func1 has been evaluated
False
xlara@lovelace:~/Documents/pls/prg2>
```

C Shell Test:

```
# Xiana Lara
# September 9th, 2020
# Programming 2: Short Circuit Evaluation for C Shell

# This program tests C Shell to see if the language implementation
# has short circuit evaluation in the AND Boolean construct.
# If C Shell is short circuited it evaluates only the first function
# to determine if the expression is true or false.

# Input:  None
# Output: 2 different and expressions
# Expression 1: False (func1 && func2)
```

```
# Expression 2: False (func2 && func1)
#!/bin/sh

func1(){

    # print statement
    echo "func1 has been evaluated"
    return 0

} # of func1

func2(){

    # print statement
    echo "func2 has been evaluated"
    return 1

} # of func2

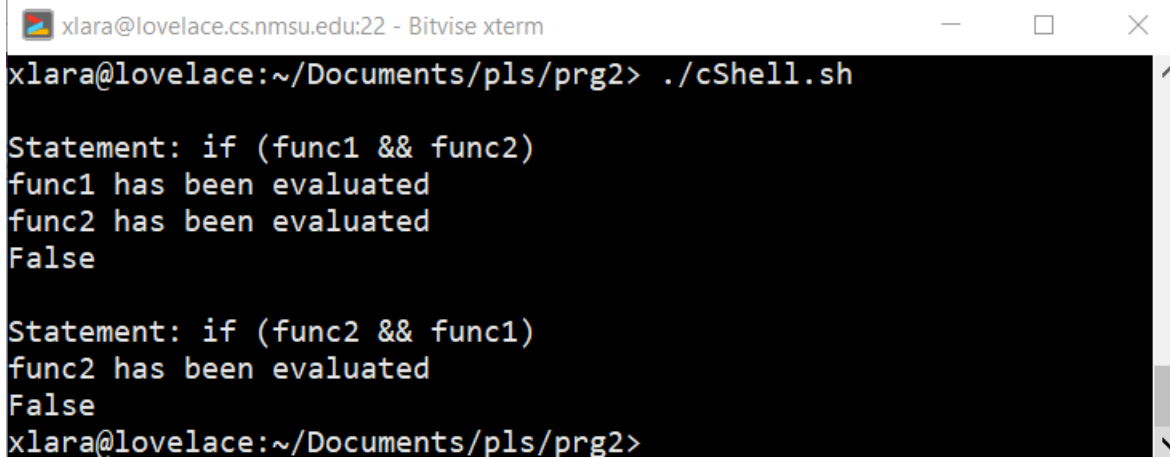
# print statement
echo
echo "Statement: if (func1 && func2)"

if (func1 && func2);
then
    # print statement
    echo "True"
else
    # print statement
    echo "False"
fi

# print statement
echo
echo "Statement: if (func2 && func1)"

if (func2 && func1)
then
    # print statement
    echo "True"
else
    # print statement
    echo "False"
fi
```

C Shell Test Results:

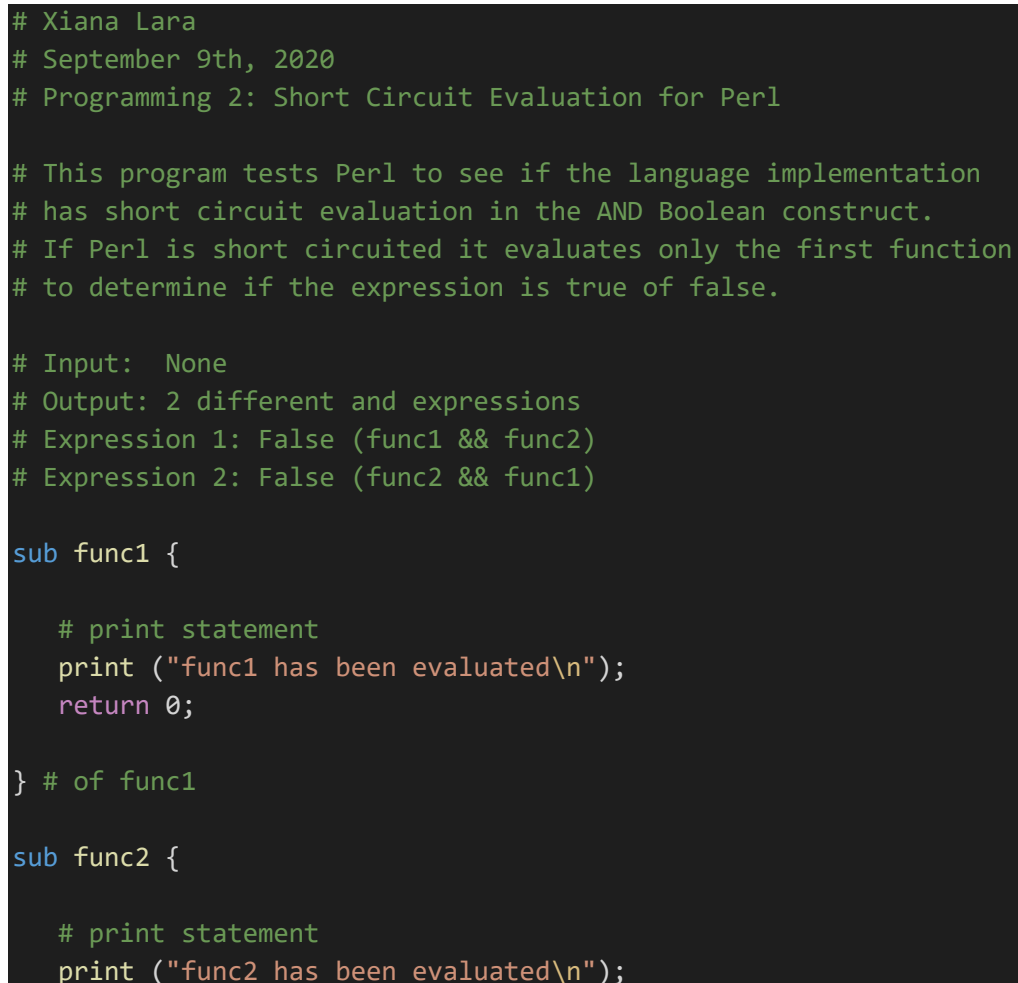


```
xlara@lovelace.cs.nmsu.edu:22 - Bitwise xterm
xlara@lovelace:~/Documents/pls/prg2> ./cShell.sh

Statement: if (func1 && func2)
func1 has been evaluated
func2 has been evaluated
False

Statement: if (func2 && func1)
func2 has been evaluated
False
xlara@lovelace:~/Documents/pls/prg2>
```

Perl Test:



```
# Xiana Lara
# September 9th, 2020
# Programming 2: Short Circuit Evaluation for Perl

# This program tests Perl to see if the language implementation
# has short circuit evaluation in the AND Boolean construct.
# If Perl is short circuited it evaluates only the first function
# to determine if the expression is true or false.

# Input:  None
# Output: 2 different and expressions
# Expression 1: False (func1 && func2)
# Expression 2: False (func2 && func1)

sub func1 {

    # print statement
    print ("func1 has been evaluated\n");
    return 0;

} # of func1

sub func2 {

    # print statement
    print ("func2 has been evaluated\n");
```

```
    return 1;

} # of func2

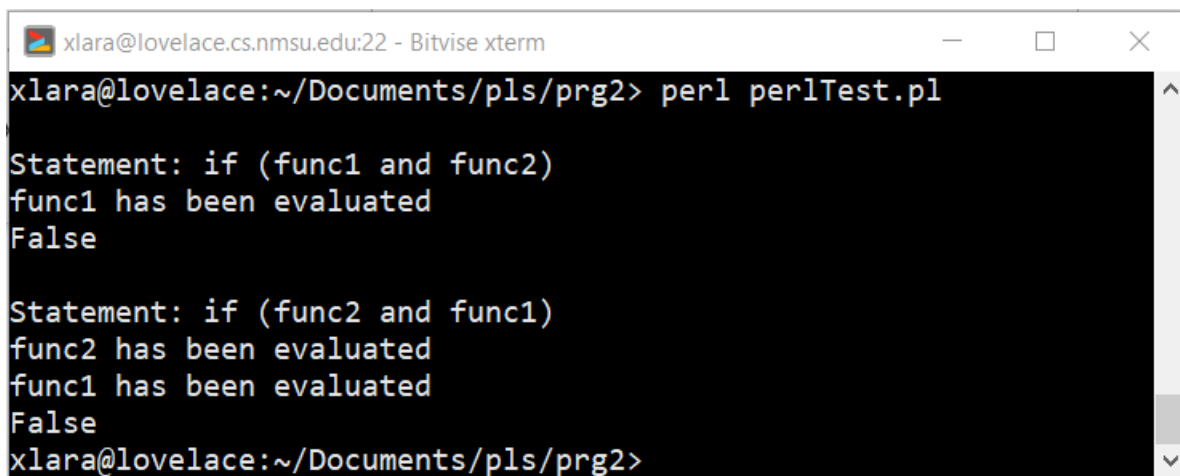
# print statement
print("\nStatement: if (func1 and func2)\n");

if(func1 and func2){
    # print statement
    print ("True\n");
}
else{
    # print statement
    print ("False\n");
}

# print statement
print("\nStatement: if (func2 and func1)\n");

if (func2 and func1){
    # print statement
    print ("True\n");
}
else{
    # print statement
    print ("False\n");
}
```

Perl Test Results:



The screenshot shows a terminal window titled "xlara@lovelace.cs.nmsu.edu:22 - Bitwise xterm". The user has executed the command "perl perlTest.pl". The output of the script is as follows:

```
xlara@lovelace:~/Documents/pls/prg2> perl perlTest.pl

Statement: if (func1 and func2)
func1 has been evaluated
False

Statement: if (func2 and func1)
func2 has been evaluated
func1 has been evaluated
False
xlara@lovelace:~/Documents/pls/prg2>
```

PHP Test:

```
<?php
# Xiana Lara
# September 9th, 2020
# Programming 2: Short Circuit Evaluation for PHP

# This program tests PHP to see if the language implementation
# has short circuit evaluation in the AND Boolean construct.
# If PHP is short circuited it evaluates only the first function
# to determine if the expression is true or false.

# Input:  None
# Output: 2 different and expressions
# Expression 1: False (func1 && func2)
# Expression 2: False (func2 && func1)

function func1(){

    # print statement
    echo "func1 has been evaluated\n";
    return 0;

} # of func1

function func2(){

    # print statement
    echo "func2 has been evaluated\n";
    return 1;

} # of func2

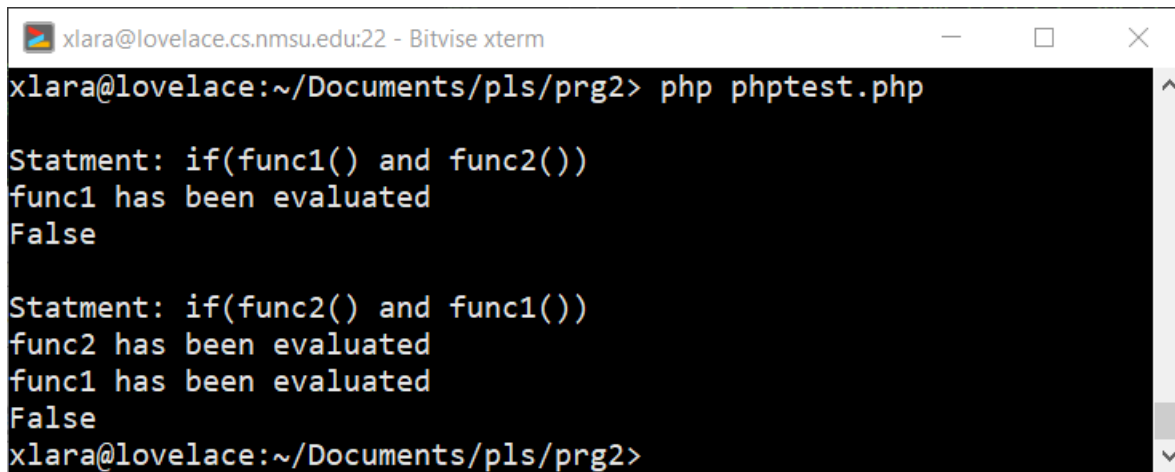
# print statement
echo "\nStatment: if(func1() and func2())\n";

if(func1() && func2()){
    # print statement
    echo "True\n";
} else {
    # print statement
    echo "False\n";
}
```

```
# print statement
echo "\nStatment: if(func2() and func1())\n";

if(func2() && func1()){
    # print statement
    echo "True\n";
} else {
    # print statement
    echo "False\n";
}
```

PHP Test Results:



The screenshot shows a terminal window titled "xlara@lovelace.cs.nmsu.edu:22 - Bitwise xterm". The user has executed the command "php phptest.php". The output of the script is as follows:

```
xlara@lovelace:~/Documents/pls/prg2> php phptest.php

Statment: if(func1() and func2())
func1 has been evaluated
False

Statment: if(func2() and func1())
func2 has been evaluated
func1 has been evaluated
False
xlara@lovelace:~/Documents/pls/prg2>
```