Jeffrey Lansford

9/92020

Program 2

We are trying to see how different languages implement Short Circuit Evaluation on the AND Boolean construct. The four programs test the AND Boolean by doing an If statement on two expressions, one where it will evaluate to false, A, and another where it calls a function to print to screen and return something to evaluate to true, B. We see if when A is evaluated first, do we see the print statement inside the B function. Then we flip A and B around, we then evaluate B first and then A and see if we get the print statement. If we do not see it in the first run and see it in the flipped second run, then the AND Boolean construct is short circuited in that language.

|  |  |  |
| --- | --- | --- |
| Language | Short Circuit | Not Short Circuit |
| Ada | Yes | Yes |
| C-Shell | Yes |  |
| Perl | Yes |  |
| PHP | Yes |  |

Ada has both a short circuit implantation and a non-short circuit implementation. The statement ‘and’ does not do short circuit and will evaluate both A and B. the statement ‘and then’ will short circuit and will not evaluate B when A is false.

Ada:

-- Jeffrey Lansford

-- 9/9/2020

-- Program 2

-- Ada program to test the short circuit implementation of Ada

with Ada.Text\_IO; use Ada.Text\_IO;

with Ada.Integer\_Text\_IO; use Ada.Integer\_Text\_IO;

procedure Pro2ada is

    A : Integer;

    -- function to see if it is executed during short circut evaluation

    function B return Integer is

    begin

        Put\_Line("      Within B");

        return 1;

    end B;

begin

    Put\_Line("A is always False and B is always True");

    Put\_Line("Here are the two conditions with 'and'");

    A := 0;

    -- Do Short Circuit Evaluation and 'and' statement in Ada

    -- A will evaluate to false and B will evaluate to true

    Put\_Line("A && B");

    if A = 1 and B = 1 then

        Put\_Line("True");

    else

        Put\_Line("False");

    end if;

    -- B will evaluate to false and A will evaluate to true

    Put\_Line("B && A");

    if B = 1 and A = 1 then

        Put\_Line("True");

    else

        Put\_Line("False");

    end if;

    -- Do Short Circuit Evaluation and 'and then' statement in Ada

    Put\_Line("");

    Put\_Line("Here are the two conditions with 'and then' with short circuting");

    Put\_Line("A && B");

    if A = 1 and then B = 1 then

        Put\_Line("True");

    else

        Put\_Line("False");

    end if;

    -- B will evaluate to false and A will evaluate to true

    Put\_Line("B && A");

    if B = 1 and then A = 1 then

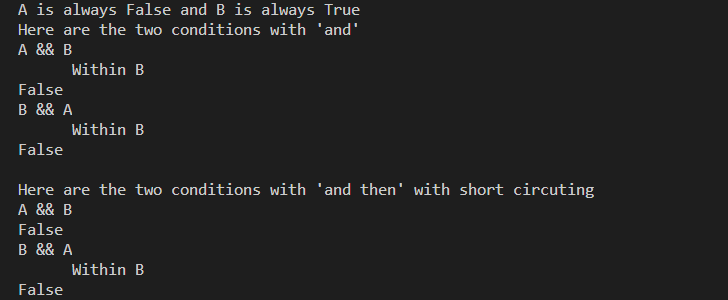
        Put\_Line("True");

    else

        Put\_Line("False");

    end if;

end Pro2ada;



C-Shell:

#!/bin/csh

# Jeffrey Lansford

# 9/9/2020

# Program 2

# C-Shell program to test the short circut implementation of C-Shell

echo "A is always False and B is always True"

set a = 0

# Short Circuit evaluation on 'and' in C-Shell

# C-Shell does not have function, so we can do call commands within the curly braces

# A evaluate to false and command B returns with exit code of 0 to evaluate to true

echo "A && B"

if (  $a == 1 && { echo "        Within B" }) then

    echo "True"

else

    echo "False"

endif

# command B returns with exit code of 0 to evaluate to true and A evaluate to false

echo "B && A"

if (  { echo "        Within B" } && $a == 1) then

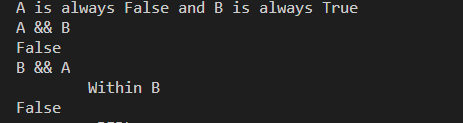
    echo "True"

else

    echo "False"

endif

exit



Perl:

#!/usr/bin/perl

# Jeffrey Lansford

# 9/9/2020

# Program 2

# Perl program to test the short circuit implementation of Perl

use strict;

use warnings;

print "A is always False and B is always True\n";

# Function to show evaluation of B in evaluations

sub b {

    print "\twithin b\n";

    return 1;

}

# Set A to value to compare with to equate 'False'

$a = 1;

# Short Circuit evaluation on 'and' in Perl

# evaluate A to false and run B() and return 1 to evaluate to true

print "A && B\n";

if ($a == 0 && b()) {

    print "True\n";

}

else{

    print "False\n";

}

# run B() and return 1 to evaluate to true and evaluate A to false

print "B && A\n";

if ( b() && $a == 0) {

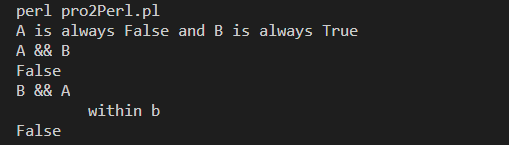
    print "True\n";

}

else{

    print "False\n";

}



PHP:

<?php

// Jeffrey Lansford

// 9/9/2020

// Program 2

// PHP program to test the short circuit implementation of PHP

echo "A is always False and B is always True\n";

// B function for testing if it is evaluated within and statement

function b () {

    echo "\twithin b\n";

    return true;

}

$a = false;

# Short Circuit evaluation on 'and' in PHP

# evaluate A to false and run B() and evaluate to true

echo "A && B\n";

if ($a && b()) {

    echo "True\n";

}

else {

    echo "False\n";

}

# run B() and evaluate to true and evaluate A to false

echo "B && A\n";

if (b() && $a ) {

    echo "True\n";

}

else {

    echo "False\n";

}

?>

