Joey Troyer
09/07/2022
Lab 2

Problem description:

The task is to evaluate four different programming languages to see if they implement short-circuit functionality. Short circuit evaluation tests how efficiently the programming language handles the AND and OR operators. For example, in the code if (false && true), you would want the program to exit the if statement as soon as it sees false instead of also evaluating the right-hand side because the condition can't possibly be true at this point. A similar scenario is true for OR. Looking at the code snippet, if (true || false) the left-hand side is already true, there is no point in evaluating the right-hand side. To test this, I wrote a program in Ada, c-shell, PHP, and Perl. Each program tests the AND and OR operators for short circuit evaluation. The results have been written below in tabular form, and screenshots of the code are below.

ADA	ADA does not have short circuit for AND nor OR. However, Ada does have short circuit for AND THEN and OR ELSE. These are the dedicated short circuit operators, while the traditional ones are logical operators.
C-SHELL	C-Shell has short circuit evaluation for both && and ||. They do have bitwise operators & and | which do not have short circuit evaluation.
PHP	PHP has short circuit evaluation for both && and || aswell as AND and OR. There are also non short circuit operators such as | and &
PERL	Perl has short circuit evaluation for both && and || aswell as AND and OR. There are also non short circuit operators such as | and &

ADA

```ada
--  Name: Joey Troyer
--  Date: 09/07/22
--  pre: None
--  post: Output to the termial

with Ada.Text_IO; use Ada.Text_IO;
with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;

procedure adaTest is

    --test function to be called
    function func return Boolean is
    begin
        Put_Line("This function was called");
        return true;
    end func;

    a : Integer;
    begin

--start of short circuit operators
        a := 0;
        Put_Line("AND evaluation using short circuit operators");
        while a < 2 loop
        --left hand side starts false then next loop is true
        if a > 0 and then func then
            Put_Line("True");
        else
            put_line("false");
        end if;

        Put_Line("  Next");
        a := a + 1;
        end loop;

        Put_Line("");

        a := 0;
        Put_Line("OR evaluation using short circuit operators");
        while a < 2 loop
        --left hand side starts false then next loop is true
        if a > 0 or else func then
            Put_Line("True");
        else
            put_line("false");
        end if;
```

```
        Put_Line("  Next");
        a := a + 1;
        end loop;


-- start of logical operators
        Put_Line("");
        a := 0;

        Put_Line("AND evaluation");
        while a < 2 loop
        --left hand side starts false then next loop is true
        if a > 0 and func then
            Put_Line("True");
        else
            put_line("false");
        end if;

        Put_Line("  Next");
        a := a + 1;
        end loop;

        Put_Line("");

        a := 0;
        Put_Line("OR evaluation");
        while a < 2 loop
        --left hand side starts false then next loop is true
        if a > 0 or func then
            Put_Line("True");
        else
            put_line("false");
        end if;

        Put_Line("  Next");
        a := a + 1;
        end loop;

end adaTest;
```

```
jtroyer@granville:~/CS471/Lab2/ADA> ./adatest
AND evaluation using short circuit operators
false
  Next
This function was called
True
  Next

OR evaluation using short circuit operators
This function was called
True
  Next
True
  Next

AND evaluation
This function was called
false
  Next
This function was called
True
  Next

OR evaluation
This function was called
True
  Next
This function was called
True
  Next
jtroyer@granville:~/CS471/Lab2/ADA>
```

C-SHELL

```
#Name: Joey Troyer
#Date: 09/02/22
#pre: none
#post: Output to the termial
#!/bin/csh

echo 'C-SHELL Short circuit evaluation'

#start of logical operators
echo 'AND test using &&'

    false && echo "foo"
    echo " next"
    true && echo "foo"

echo ""

echo 'OR test using ||'
    false || echo "foo"
    echo " next"
    true || echo "foo"

#bitwise and
echo 'AND test using &'

    false & echo "foo"
    echo " next"
    true & echo "foo"

echo ""

#bitwise or
echo 'OR test'
    false | echo "foo"
    echo " next"
    true | echo "foo"
```

```
jtroyer@granville:~/CS471/Lab2/cshell> csh shelltest.csh
C-SHELL Short circuit evaluation
AND test using &&
 next
foo

OR test using ||
foo
 next

AND test using &
[1] 3871
foo
 next
[2] 3872
foo

OR test
foo
 next
foo
[1]    Exit 1                        false
jtroyer@granville:~/CS471/Lab2/cshell>
```

PHP

```php
<?php
// file phptest.php
// author Joey Troyer
// date 2022-09-01
// pre none
// post output to the terminal


    // test function to be called
    function func() {
        echo "I have been evaluated!!!\n";
        return 1;
    }//end func

    //start of short circuit operators
    $var = 1;

    echo "AND evaluation using &&\n";
    for($i = 0; $i < 2; $i++) {
        //left hand side starts false then next loop is true
        if( $i == 1 && func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for

    echo "\n";

    echo "OR evaluation using ||\n";
    for($i = 0; $i < 2; $i++) {
        //left hand side starts false then next loop is true
        if( $i == 1 || func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for


    //start of other short circuit operators
    $var = 1;

    echo "AND evaluation using and\n";
    for($i = 0; $i < 2; $i++) {
```

```php
        //left hand side starts false then next loop is true
        if( $i == 1 and func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for

    echo "\n";

    echo "OR evaluation using ||\n";
    for($i = 0; $i < 2; $i++) {
        //left hand side starts false then next loop is true
        if( $i == 1 or func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for


    //start of eager operators
    $var = 1;

    echo "AND evaluation using &\n";
    for($i = 0; $i < 2; $i++) {
        //left hand side starts false then next loop is true
        if( $i == 1 & func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for

    echo "\n";

    echo "OR evaluation using |\n";
    for($i = 0; $i < 2; $i++) {
        //left hand side starts false then next loop is true
        if( $i == 1 | func())
            echo "true\n";
        else
            echo "false\n";

        echo "  next\n";
    }//end for
?>
```

```
jtroyer@granville:~/CS471/Lab2/PHP> php phptest.php
AND evaluation using &&
false
  next
I have been evaluated!!!
true
  next

OR evaluation using ||
I have been evaluated!!!
true
  next
true
  next
AND evaluation using and
false
  next
I have been evaluated!!!
true
  next

OR evaluation using ||
I have been evaluated!!!
true
  next
true
  next
AND evaluation using &
I have been evaluated!!!
false
  next
I have been evaluated!!!
true
  next

OR evaluation using |
I have been evaluated!!!
true
  next
I have been evaluated!!!
true
  next
```

Perl

```perl
# file perltest.pl
# author Joey Troyer
# date 2022-09-07
# pre none
# post output to the terminal

#test functin to be called
sub func() {
    print "I have been evaluated!!!\n";
    return 1;
}# end func

#start of short circuit operators
$var = 1;
print "\nAND evaluation using short circuit operator &&\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 && func()) {
        print "true\n";
    }else {
        print "false\n";
    }

    print "  next\n";
}# end for

print "\n";
print "OR evaluation using short circuit operator ||\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 || func()) {
        print "true\n";
    }else {
        print "false\n";
    }

    print "  next\n";
}# end for


#start of other short circuit operators
$var = 1;
print "\nAND evaluation using and\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 and func()) {
        print "true\n";
    }else {
```

```
        print "false\n";
    }

    print "  next\n";
}# end for

print "\n";
print "OR evaluation using or\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 or func()) {
        print "true\n";
    }else {
        print "false\n";
    }

    print "  next\n";
}# end for


#start of logical operators
$var = 1;
print "\nAND evaluation using &\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 & func()) {
        print "true\n";
    }else {
        print "false\n";
    }

    print "  next\n";
}# end for

print "\n";
print "OR evaluation using |\n";
for($i = 0; $i < 2; $i++) {
    #left hand side starts false then next loop is true
    if( $i == 1 | func()) {
        print "true\n";
    }else {
        print "false\n";
    }

    print "  next\n";
}# end for
```

```
jtroyer@granville:~/CS471/Lab2/perl> perl perltest.pl

AND evaluation using short circuit operator &&
false
  next
I have been evaluated!!!
true
  next

OR evaluation using short circuit operator ||
I have been evaluated!!!
true
  next
true
  next

AND evaluation using and
false
  next
I have been evaluated!!!
true
  next

OR evaluation using or
I have been evaluated!!!
true
  next
true
  next

AND evaluation using &
I have been evaluated!!!
false
  next
I have been evaluated!!!
true
  next

OR evaluation using |
I have been evaluated!!!
true
  next
I have been evaluated!!!
true
  next
jtroyer@granville:~/CS471/Lab2/perl>
```