PA 2: Short Circuit Evaluation

<u>Name:</u> Indronil Bhattacharjee

<u>Problem Description:</u>
The goal of this assignment is to test whether the programming languages ADA, Bash Shell, PHP, and PERL implement short circuit evaluation in the AND and OR Boolean constructs. Short circuit evaluation is when the language evaluates the first portion of a BOOLEAN expression and if, knowing the result of the value, then skips the evaluation of the second expression.

<u>Summary of Results:</u>

Language	AND Short Circuit
ADA	No
Bash Shell	Yes
PHP	Yes
PERL	Yes

Three of the programming languages have implemented short circuit evaluation in the AND Boolean constructs except ADA.

<u>Program 1: Short Circuit Evaluation in ADA</u> (Referring to *shortcircuit_Ada.adb* file)

ADA

```
with Ada.Text_IO;

procedure Main is
   function F return Boolean is
   begin
      Ada.Text_IO.Put_Line("I have been evaluated");
      return True;
   end F;

   I : Integer := 1;

begin
   if I = 0 and then F then
      Ada.Text_IO.Put_Line("True");
   else
      Ada.Text_IO.Put_Line("False");
   end if;
end Main;
```

<u>Output:</u>

ADA

```
I have been evaluated
False
```

<u>Program 2: Short Circuit Evaluation in Bash Shell</u> (Referring to *shortcircuit_Bash.bash* file)

Bash

```bash
function f() {
  echo "I have been evaluated"
  return 1
}

i=1
if [[ $i == 0 ]] && f; then
  echo "True"
else
  echo "False"
fi
```

<u>Output:</u>

Bash

```
False
```

<u>Program 3: Short Circuit Evaluation in PHP</u> (Referring to *shortcircuit_PHP.php* file)

PHP

```php
<?php
function f() {
  echo "I have been evaluated";
  return 1;
}

$i = 1;

if ($i == 0 && f()) {
  echo "True";
}
else {
  echo "False";
}
?>
```

<u>Output:</u>

PHP

```
False
```

Program 4: Short Circuit Evaluation in Perl (Referring to *shortcircuit_Perl.pl* file)

```perl
sub f {
  print "I have been evaluated";
  return 1;
}

$i = 1;

if ($i == 0 && f()) {
  print "True";
}
else {
print "False";
}
```

Output:

```
False
```

Short circuit evaluation occurs when the language evaluates the first portion of a Boolean expression and, if the result is sufficient to determine the overall value of the expression, it skips the evaluation of the remaining portion of the expression.

In this case, the AND operator is used, which means that both conditions must be true in order for the overall expression to be true. Therefore, if i is equal to 0, then the overall expression will be false, regardless of the value returned by the f() function. In this case, short circuit evaluation will cause the function f() to never be evaluated since the first part of the expression is already false.

Therefore, for 3 of the 4 languages above, short circuit evaluation occurs and prints **"False"** for the code output except ADA. Bash Shell, PHP and Perl have implemented short circuit evaluation in the AND Boolean constructs except ADA.