Melis Atun
21901865
CS 315 - Section 01
Homework II

DART

1. How are the boolean values represented?

   bool melis = 2 > 1;

   print(melis); //true

   bool boolean = 1 > 3;

   print(boolean); //false

   bool myName = true;

   if (myName) {

   print("Your name is Melis.");

   }

   else {

   print("You have no name.");

   }

2. What operators are short-circuited?

   - && and || operators are short-circuited -

   var m = 2;

var k = 4;

var test = (m < k) && (k > m);

print(test); //true

var i = 10;

var j = 50;

var myNum = (j > 60) || (i > 60);

print(myNum); //false

var myOtherNum = (j > i) || (i > j);

print(myOtherNum); //true

3. How are the results of short-circuited operators computed? (Consider also function calls)

   The && operator first checks the first operand. If the first operand is false, it returns false without checking the second operand. However, if the first operand is true and the second operand is false, it will return false (meaning that it will check the second operand).

   var melo = 80;

   var testing = (melo > 120) && (melo < 100);

   print(testing);

   var myNo = 78;

   var testNo = (myNo > 80) && (myNo < 90);

print(testNo);

The || operator first checks the first operand. If it is true, it will not check the second operand. However, if the first operand is false, then it will check the second operand and return the result accordingly.

var myFavNum = 8;

var secondTesting = (myFavNum > 5) || (myFavNum < 3);

print(secondTesting);

var thirdTesting = (myFavNum < 5) || (myFavNum > 4);

print(thirdTesting);

```
bool functionName() {
    print("Testing function…");
    return true;
}
```

```
print((myFavNum < 5) || functionName());
print((myFavNum > 4) && functionName());
print((myFavNum > 4) || functionName());
print((myFavNum < 5) && functionName());
print((myFavNum < 12) && (myFavNum < 4));
```

4. What are the advantages of short-circuit evaluation?

Short-circuit evaluation can save a programmer from doing an unnecessary number of computations while coding. Also, it prevents a programmer from having "out-of-bounds" error which can be extremely painful to find where the error occurs.

var myArray = [0, 3, "CS 315" , "homework"];

var myNumber = 0;

while (myNumber < 4 && myArray[myNumber] != "homework") {

print(myArray[myNumber]);

myNumber++;

}

5. What are the potential problems with short-circuit evaluation?

If a function will be executed after the program has been run by the programmer, the function may not be executed because of the short-circuiting existing in the program.

var myFather = false && functionName(); //function will not be called and executed because first operand is false

print(myFather);

var myMother = true || functionName(); //function will not be called and executed again because the first operand is true

print(myMother);

JAVASCRIPT

1. How are the boolean values represented?
   - **with the boolean() function**

   console.log(Boolean(6 > 2)); //true

   let m = 0;

   console.log(Boolean(m)); //false

   let k = 1;

   console.log(Boolean(k)); //true

   let i = -0;

   console.log(Boolean(i)); //false

   let n = " ";

   console.log(Boolean(n)); //false

   let melis;

   console.log(Boolean(melis)); //false

   let melo = null;

   console.log(Boolean(melo)); //false

   let bool = false;

   console.log(Boolean(bool)); //false

2. What operators are short-circuited?

   - && and || operators are short-circuited -

   console.log(true && false); //false

console.log(true && true); //true

console.log(true || false); //true

console.log(true || true); //true

console.log(false && true); //false

console.log(false && false); //false

console.log(false || true); //true

console.log(false || false); //false

3. How are the results of short-circuited operators computed? (Consider also function calls)

   The && operator returns true if and only if both of the operands are true. It returns false in any other condition.

   let a = 1;

   let b = 1;

   let c = a && b;

   let d = 0;

   console.log(c); //1

   console.log(a && d); //0

   console.log(b && d); //0

   The || operator returns true if and only if one or more of its operands are true. It returns false in any other condition.

   console.log(a || b); //1

console.log(a || d); //1

console.log(b || d); //1

4. What are the advantages of short-circuit evaluation?

   Short-circuit evaluation can save a programmer from doing an unnecessary number of computations while coding. Also, it prevents a programmer from having "out-of-bounds" error which can be extremely painful to find where the error occurs.

   let myArray = [0, 3, "CS 315", "homework"];

   let myNumber = 0;

   while (myNumber < 4 && myArray[myNumber] != "homework") {

   console.log(myArray[myNumber]);

   myNumber = myNumber + 1;

   }

5. What are the potential problems with short-circuit evaluation?

   If a function will be executed after the program has been run by the programmer, the function may not be executed because of the short-circuiting existing in the program.

   let myFather = false && functionName(); //function will not be called and executed because first operand is false

   console.log(myFather); //false

let myMother = true || functionName(); //function will not be called and executed again because the first operand is true

console.log(myMother); //true

PHP

1. How are the boolean values represented?

   $melo = true;

   $melis = false;

   var_dump($melo); //true

   var_dump($melis); //false

   $myBool = 0;

   var_dump($myBool); //0

   $try = (5 > 4);

   var_dump($try); //true

2. What operators are short-circuited?

   - &&, ||, AND, OR operators are short-circuited -

   function functionName() {

   echo ("Trying boolean values…");

   return true;

   }

   if (false && functionName()) {

echo ("Result is…");

}

$myNumber = true;

$myString = $myNumber && (functionName() || true) ? 'True' : 'False';

var_dump($myString);

$myNum = false or true;

var_dump($myNum); //false

$myOtherNum = false and true;

var_dump($myOtherNum); //false

3. How are the results of short-circuited operators computed? (Consider also function calls)

   The && operator returns false if and only if one or more of its operands are false. It returns true if and only if one or more of its operands are true.

   $myFavNum = 8;

   $tryNum = ($myFavNum < 100 && $myFavNum < 7);

   var_dump($tryNum); //false

   $tryOtherNum = ($myFavNum > 100 && $myFavNum > 7);

   var_dump($tryNum); //false

The || operator returns true if and only if one or more of its operands are true. It returns false in any other condition.

$myMelis = ($myFavNum < 100 || $myFavNum > 7);

var_dump($myMelis); //true

$myNumber = ($myFavNum > 100 || $myFavNum > 7);

var_dump($myNumber); //true

4. What are the advantages of short-circuit evaluation?

   Short-circuit evaluation can save a programmer from doing an unnecessary number of computations while coding. Also, it prevents a programmer from having "out-of-bounds" error which can be extremely painful to find where the error occurs.

   $myArray = ["melis", "atun", "CS 315", "homework"];

   $myNumber = 0;

   while (($myNumber < 4) && ($myArray[$myNumber]) != "homework") {

   echo($myArray[$myNumber]);

   $myNumber = $myNumber + 1;

   }

5. What are the potential problems with short-circuit evaluation?

   If a function will be executed after the program has been run by the programmer, the function may not be executed because of the short-circuiting existing in the program.

$myFather = false && functionName(); //function will not be called and executed because first operand is false

var_dump($myFather); //false

$myMother = true || functionName(); //function will not be called and executed again because the first operand is true

var_dump($myMother); //true

PYTHON

1. How are the boolean values represented?

   a = 3

   b = 4

   if b > a:

   print("True.")

   else:

   print("False.")

   print(bool(b)) #true

2. What operators are short-circuited?

   - AND, OR, NOT operators are short-circuited -

   x = 8

   y = 9

   if y > x and y < 10

print("True!")

if y < x or y < 10

print("True again!")

if not x

print("Not x.")

3. How are the results of short-circuited operators computed? (Consider also function calls)

   The && operator returns false if and only if one or more of its operands are false. It returns true if and only if both of its operands are true.

   The || operator returns true if and only if one or more of its operands are true. It returns false in any other condition.

   num = 88

   myNum = num < 80 and num > 70

   print(myNum) #false

   myOtherNum = 100

   myNum = num < 200 or num > 300

   print(myNum) #true

   def functionName():

   return False

   myBool = False and functionName()

print(myBool) #false

myBool2 = True or functionName()

print(myBool2) #true

myBool3 = False or functionName()

print(myBool3) #false

myBool4 = True and functionName()

print(myBool4) #false

4. What are the advantages of short-circuit evaluation?

   Short-circuit evaluation can save a programmer from doing an unnecessary number of computations while coding. Also, it prevents a programmer from having "out-of-bounds" error which can be extremely painful to find where the error occurs.

   myArray = ["melis", "atun", "CS 315", "homework"];

   myNumber = 0;

   while ((myNumber < 4) and (myArray[myNumber]) != "homework"):

   print(myArray[myNumber]);

   myNumber = myNumber + 1;

5. What are the potential problems with short-circuit evaluation?

   If a function will be executed after the program has been run by the programmer, the function may not be executed because of the short-circuiting existing in the program.

myFather = false and functionName(); #function will not be called and executed because first operand is false

print(myFather); #false

myMother = true or functionName(); #function will not be called and executed again because the first operand is true

print(myMother); #true

RUST

1. How are the boolean values represented?

   let a: bool = true;

   let b: bool = false;

   println!("First result is: {} ", a == b); //false

   println!("Second result is: {} ", a | b); //true

   println!("Third result is: {} ", a & b); //false

   let c = 4;

   let d = 5;

   println!("Fourth result is: {} ", d > c); //true

2. What operators are short-circuited?

   - && and || operators are short-circuited -

   let x = true || false;

   let y = true && false;

let z = false || true;

let k = false && true;

println!("1st result is: {} ", x); //true

println!("2nd result is: {} ", y); //false

println!("3rd result is: {} ", z); //true

println!("4th result is: {} ", k); //false

3. How are the results of short-circuited operators computed? (Consider also function calls)

   The && operator returns false if the first operand is false and it does not check the second operand. However, if the first operand is true and the second operand is false, it returns false (meaning that it checks the second operand). It returns true if and only if both of its operands are true.

   The || operator returns true if and only if one or more of its operands are true without regarding the order (meaning that if the first operand is true or the second operand is true or both of them are true, does not matter). It returns false in any other condition.

   let m = 50;

   let i = 80;

   let my_bool = m < 40 && i > 70;

   println!("Result 1: ", my_bool); //false

   let my_bool2 = m > 40 && i < 70;

```
println!("Result 2: ", my_bool2); //false

let my_bool3 = m < 40 || i > 70;

println!("Result 3: ", my_bool3); //true

let my_bool4 = m > 40 || i < 70;

println!("Result 4: ", my_bool4); //true
```

4. What are the advantages of short-circuit evaluation?

Short-circuit evaluation can save a programmer from doing an unnecessary number of computations while coding. Also, it prevents a programmer from having "out-of-bounds" error which can be extremely painful to find where the error occurs.

```
let my_array = ["melis", "atun", "CS 315", "homework"];

let mut my_number = 0;

fn function_name() -> bool {

    println!("If this is printed, then no short circuit.");

    true

}

while my_number < 4 && my_array[my_number] !=
"homework" {

    println!("Array: ", my_array[my_number]);

    my_number += 1;

}
```

5. What are the potential problems with short-circuit evaluation?

   If a function will be executed after the program has been run by the programmer, the function may not be executed because of the short-circuiting existing in the program.

   let my_father = false && function_name(); //function will not be called and executed because first operand is false

   println!("Result: ", my_father); //false

   let my_mother = true || function_name(); //function will not be called and executed again because the first operand is true

   println!("Result: ", my_mother); //true

Which language is the best for short-circuit evaluation?

In my opinion, Python is the best programming language for short-circuit evaluation. First of all, it is the most readable language that I have ever seen, which makes it very easy to code in Python. It is not complicated at all. Furthermore, unlike other programming languages, the && operator returns true if and only if both of its operands are true. This means that Python checks the second operand as well as the first one, which makes it a reliable programming language and paves the way for avoiding unnecessary and complicated errors. For example, some other language (like Rust) does not check the second operand and returns false immediately if the first operand is false, which is not something realiable in a programming language. Also, the || operator returns true if and only if one or more of its operands are true, which is again a proof of Python's reliability. Python also supports value evaluation and >, < operators for the evaluation of boolean expressions. For example, it allows to print the result of (a < b) or (b > a) as a boolean value, which is not allowed in some other programming languages. In addition,

Python supports representing strings, integers, and other primitive types in boolean as well.

My Learning Strategy

First of all, without starting to write any code snippets, I did a general research on the programming language's short-circuit evaluation that I am studying at that moment. This allowed me to have a general information about that language's short-circuit evaluation, which operators are used for it, etc. and made me familiar with the subject in general. After doing this brief research, I started to research the specific five questions that we have in the homework. I tried to write code snippets for each question rather than writing a whole program and this allowed me to learn the rules of the programming languages' short-circuit evalution better without mixing them up and being drowned in errors later on. I tried to look into every resource that I can found so that I can have a better understanding of the specific rule that will be the answer of a question. However, I also encountered some unreliable resources that lack necessary information and have redundant information and I eliminated them. Later on, after writing code snippets for every question, I brought them together in a whole program and put the program in Atom text editor, which is my favorite one to use. My teacher David told me about Atom. After saving my program with the proper extension (.html, .rs, .py, .dart, .php), I opened terminal on my MacBook and ran my program. At first, I encountered many errors which I inspected of course but the errors were less this time compared to the first homework. Also, solving the errors took less time this time. Then, I studied the errors that I got by googling them or simply correcting them if they are simple mistakes that I did not notice (for example, accidentally writing ; at the end of lines in Python or forgetting to put the closing parenthesis). After I resolved all of the errors, I opened FileZilla and connected to the school server. Since I am connected to school WIFI because I am staying at dorms, I did not have to use VPN which made my life a lot easier because VPN works very slowly on my computer. Then, I uploaded my program to the server and tested it again on Terminal (with ssh,

I do not need Putty on MacOS). However, I want to note that Javascript cannot be tested like the other four languages on Dijkstra or Mac Terminal. Therefore, I opened Vscode and I put my Javascript file on Vscode and ran it. When I clicked on the run button, Google Chrome opened up and there it was, my program. However, to be able to see the output and the errors if any on the console, I right clicked on the page and clicked on “Inspect”, then I clicked on “console”. I checked my program from the console and resolved the mistakes if I had any. After being completely sure that my code works properly without any errors and it is a clean code with proper comments and everything, I closed the program. Then, I checked my pdf report of the homework because I made many changes on the code snippets since I resolved many errors and added some stuff in order to test my program better. After I made sure that my pdf report is proper as well, I moved on to the next language. I repeated this process for every five languages in the homework.

References

"An essential guide to PHP boolean," *PHP Tutorial*, 25-Jun-2021. [Online]. Available: https://www.phptutorial.net/php-tutorial/php-boolean/. [Accessed: 06-Dec-2021].

B. Morelli, "JavaScript - Short Circuit conditionals," *Medium*, 28-Nov-2017. [Online]. Available: https://codeburst.io/javascript-short-circuit-conditionals-bbc13ac3e9eb. [Accessed: 06-Dec-2021].

Dart programming - boolean. [Online]. Available: https://www.tutorialspoint.com/dart_programming/dart_programming_boolean.htm. [Accessed: 06-Dec-2021].

Dart programming - logical operators. [Online]. Available: https://www.tutorialspoint.com/dart_programming/dart_programming_logical_operators.htm. [Accessed: 06-Dec-2021].

"Does PHP have short-circuit evaluation?," *NewbeDEV*. [Online]. Available: https://newbedev.com/does-php-have-short-circuit-evaluation. [Accessed: 06-Dec-2021].

JavaScript booleans. [Online]. Available: https://www.w3schools.com/js/js_booleans.asp. [Accessed: 06-Dec-2021].

"Javascript short circuiting operators," *GeeksforGeeks*, 11-Sep-2020. [Online]. Available: https://www.geeksforgeeks.org/javascript-short-circuiting/. [Accessed: 06-Dec-2021].

Python booleans. [Online]. Available: https://www.w3schools.com/python/python_booleans.asp. [Accessed: 06-Dec-2021].

Short circuit evaluation. [Online]. Available: https://pythoninformer.com/python-language/intermediate-python/short-circuit-evaluation/. [Accessed: 06-Dec-2021].

"The rust reference," *Types - The Rust Reference*. [Online]. Available: https://doc.rust-lang.org/reference/types.html. [Accessed: 06-Dec-2021].