CS315 HOMEWORK 2

Alp Tuğrul Ağçalı

21801799

Section 3

DART

```dart

void main() {
  for(var test = 0; test < 10; test++){
    print("test = $test");
    if(test==3) break;
  }

  for( var outer = 0 ; outer < 10; outer++ ) {
    print("outer = $outer");
    for(var j = 0; j <10; j++){
       if(outer==2) break;
    }
  }
  print("---------------");
  label:
  for( var outer = 0 ; outer < 10; outer++ ) {
    print("outer = $outer");
    for(var j = 0; j <10; j++){
       if(outer==2) break label;
    }
  }
}
```

```
test = 0
test = 1
test = 2
test = 3
outer = 0
outer = 1
outer = 2
outer = 3
outer = 4
outer = 5
outer = 6
outer = 7
outer = 8
outer = 9
---------------
outer = 0
outer = 1
outer = 2
```

In my investigation in Dart Language, I use three different for loops. In first for loop I checked that whether the language provide unconditional exit statements and I found that it provides with “break;” statement. In second for loop, I checked that if this break statement is exits two nested loop and I see it does not. So I searched that wheter the language has unconditional labeled exit statement, with the results of this search, I write third for loop as an example of unconditional labeled exit statement.

JavaScript

```
for(var test = 0; test < 10; test++){
    console.log("test = " + test);
    if(test === 3) break;
}

for( var outer = 0 ; outer < 10; outer++ ) {
    console.log("outer = " +outer);
    for(var j = 0; j < 10; j++){
        if(outer===2) break;
    }
}
console.log("--------------------");
label:
for( var outer = 0 ; outer < 10; outer++ ) {
    console.log("outer = " +outer);
    for(var j = 0; j < 10; j++){
        if(outer===2) break label;
    }
}
```

```
test = 0
test = 1
test = 2
test = 3
outer = 0
outer = 1
outer = 2
outer = 3
outer = 4
outer = 5
outer = 6
outer = 7
outer = 8
outer = 9
--------------------
outer = 0
outer = 1
outer = 2
```

In my investigations about JavaScript language, again, I writed three for loops. Firstly, I tested break statement, which is useful in Dart, again and I see that it works in JavaScript too. Then, in second for loop, I tried whether this break statement exits from nested loops and result was negative just like Dart. Finally, I write third for loop and try same way with the Dart for checking unconditional labeled exit statements and it worked again. As far as I learned from these investgations, Dart and JavaScript have similar syntaxes about this topic.

Lua

```lua
for test = 0,10,1
do
   print("test = ", test)
   if(test==3) then break end
end

for outer = 0, 10, 1
do
    print("outer = ", outer)
    for j = 0, 10, 1
    do
        if(outer == 2) then break end
    end
end
 print("---------------")

for outer = 0, 10, 1
do
    print("outer = ", outer)
    for j = 0, 10, 1
    do
        if(outer == 2) then goto done end
    end
end
::done::
goto skip
print("Does not skipped")
::skip::
```

In my invetstigations about Lua language, again, I writed three for loops. In first one I checked break statement and it worked. Then, same with first two language, I checked wheter the break statement exits from nested loops and I found that it does not again. So I writed third loop and tried same labeling method with first two. It gived error. So I searched that, if this language has unconditional labeled exit statements. So I found goto function with label. However, I tried that whether goto function is exit statement or only part of it, to do this I write “goto skip” and “::skip::” at the end of program and printed something between them. So, I found that This function is used for jumping lines, it is not unconditional labeled exit.

PHP

```php
<?php
    for($test = 0; $test < 10; $test++){
        echo "test = $test \n";
        if($test == 3) break;
    }
    for($outer = 0; $outer < 10; $outer++){
        echo "outer = $outer \n";
        for($j = 0; $j < 10; $j++){
            if($outer == 2) break;
        }
    }
    echo "----------------------\n";
    for($outer = 0; $outer < 10; $outer++){
        echo "outer = $outer \n";
        for($j = 0; $j < 10; $j++){
            if($outer == 2) break 2;
        }
    }

?>
```

```
test = 0
test = 1
test = 2
test = 3
outer = 0
outer = 1
outer = 2
outer = 3
outer = 4
outer = 5
outer = 6
outer = 7
outer = 8
outer = 9
----------------------
outer = 0
outer = 1
outer = 2
```

In PHP, I tried three for loops. In first one I tried break statement and it worked. In second for loop I checked whether break statement exits from nested loops and I see that it doesn't. So I researched that whether this language provides a unconditional labeled exits, and I found that in php programmer does not need labeled lines. Because, when I write nested loop amount near break statement it exits this amount of nested loop.

Phyton

```
for test in range (10):
    print ("test = ", test)
    if(test == 3): break

for outer in range (10):
    print ("outer = ", outer)
    for j in range (10):
        if(outer == 2): break

print("----------------")
breakOut = False
for outer in range (10):
    print ("outer = ", outer)
    for j in range (10):
        if(outer == 2):
            breakOut = True
            break
    if breakOut:
        break
```

```
test =  0
test =  1
test =  2
test =  3
outer =  0
outer =  1
outer =  2
outer =  3
outer =  4
outer =  5
outer =  6
outer =  7
outer =  8
outer =  9
----------------
outer =  0
outer =  1
outer =  2
```

In phton, I tried three for loops. In first one I tried break statement, and it worked again. In second for loop I tried that whether this break statement exits from two different for loops and, result was negative. So, I searched that whether thsi language provides unconditional labeled exit and I can't find any method for it. However, I found how can the programmer exits from nested loop, with using two if statements, and I tried this way.

Ruby

```ruby
for test in 1..10 do
 print "test = ",test,"\n"
 if test == 3
    break
 end
end

for outer in 1..10 do
    print "outer = ", outer,"\n"
    for j in 1..10 do
        if outer == 3
            break
        end
    end
end

puts "------------"
catch :label do
    for outer in 1..10 do
        print "outer = ", outer,"\n"
        for j in 1..10 do
            throw :label if outer == 3
        end
    end
end
```

```
test = 1
test = 2
test = 3
outer = 1
outer = 2
outer = 3
outer = 4
outer = 5
outer = 6
outer = 7
outer = 8
outer = 9
outer = 10
------------
outer = 1
outer = 2
outer = 3
```

In ruby language I writed 3 for loops. First two was same with examples before. Howewer, In third for loop I found throw and catch methods for exiting nested loops.

Rust

```rust
fn main() {
    for test in 1..10 {
        println!("test = {}", test);
        if test == 3{
            break;
        }
    }
    for outer in 1..10 {
        println!("outer = {}", outer);
        for j in 1..5 {
            if outer == 3{
                break;
            }
        }
    }
    println!("---------------");
    'label: for outer in 1..10 {
        println!("outer = {}", outer);
        for j in 1..5 {
            if outer == 3{
                break 'label;
            }
        }
    }
}
```

```
test = 1
test = 2
test = 3
outer = 1
outer = 2
outer = 3
outer = 4
outer = 5
outer = 6
outer = 7
outer = 8
outer = 9
---------------
outer = 1
outer = 2
outer = 3
```

In my investigations about Rust language, I wrote 3 for loops. First two was same with other examples. For third one I found there are unconditional labeled exit statements in rust. And I tried this in my example.

Evaluation of Languages:

In these languages, Phyton was less useful about exits beceause it does not provides unconditional labeled exits. Rust, Ruby, Lua was hard to write. Dart and JavaScript has similar Syntaxes which I found easier. These two was okay. However, I think best exit method was the one that PHP has. It is much easier to write.

Learning Strategies:

To doing these homework, I studied the syntaxes of languages form geeks4geeks and tutorialspoint. I searched the exiting methods from nested loops through stackoverflow. In JavaScript I only searched syntax and find my results through trying same methods with dart. In lua I tried that whether goto method is used for every jumping. In ruby I tried to make a loop with using try and catch but I could not.

Compilers:

-For Dart: https://dartpad.dev/?

-For JavaScript: https://www.programiz.com/javascript/online-compiler/

-For Lua: https://www.tutorialspoint.com/execute_lua_online.php

-For PHP: https://www.tutorialspoint.com/execute_php_online.php

-For Phyton: https://www.programiz.com/python-programming/online-compiler/

-For Ruby: https://www.tutorialspoint.com/execute_ruby_online.php

-For Rust: https://play.rust-lang.org/