CS 315

PROGRAMMING LANGUAGES

HW #2 REPORT

Berk Yıldız

21502040

Section 1

Table of Contents

1. JavaScript

*Details are given in example program.

- **What are the types of loop control variables?**
  - **For:** Numbers
  - **For-in:** All data types
  - **While:** Number, String and Boolean
  - **Do-while:** Numbers, Strings and Boolean
- **What are the scopes of loop control variables?**

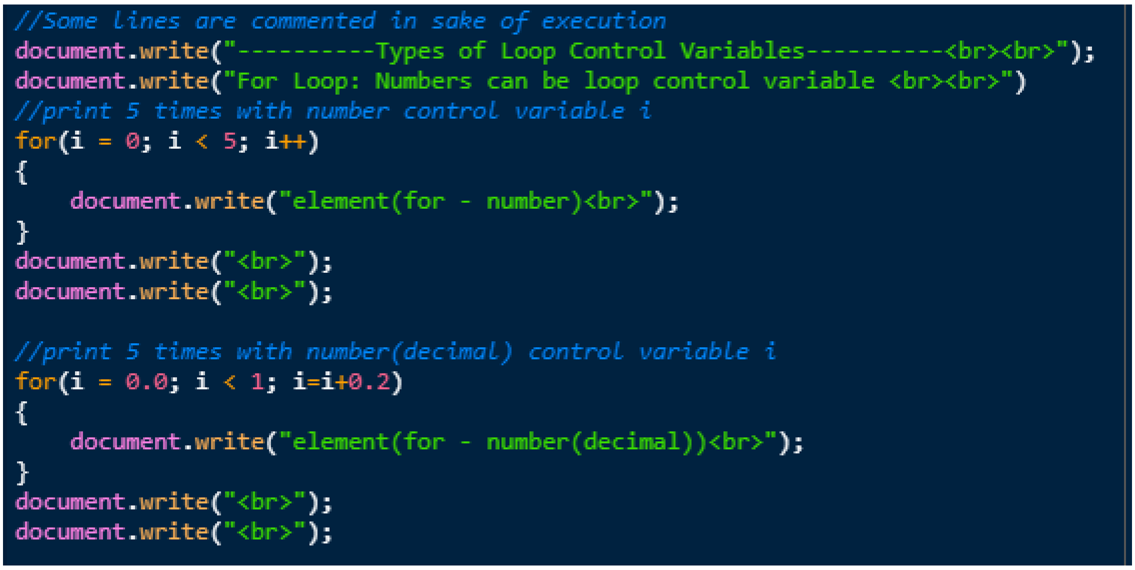
Scope of loop control variables are not limited with the loop and they act like global variable. Loop control variables are visible on all program.

- **Is it legal for the loop control variable or loop parameters to be changed in the loop, and if so, does the change affect loop control?**

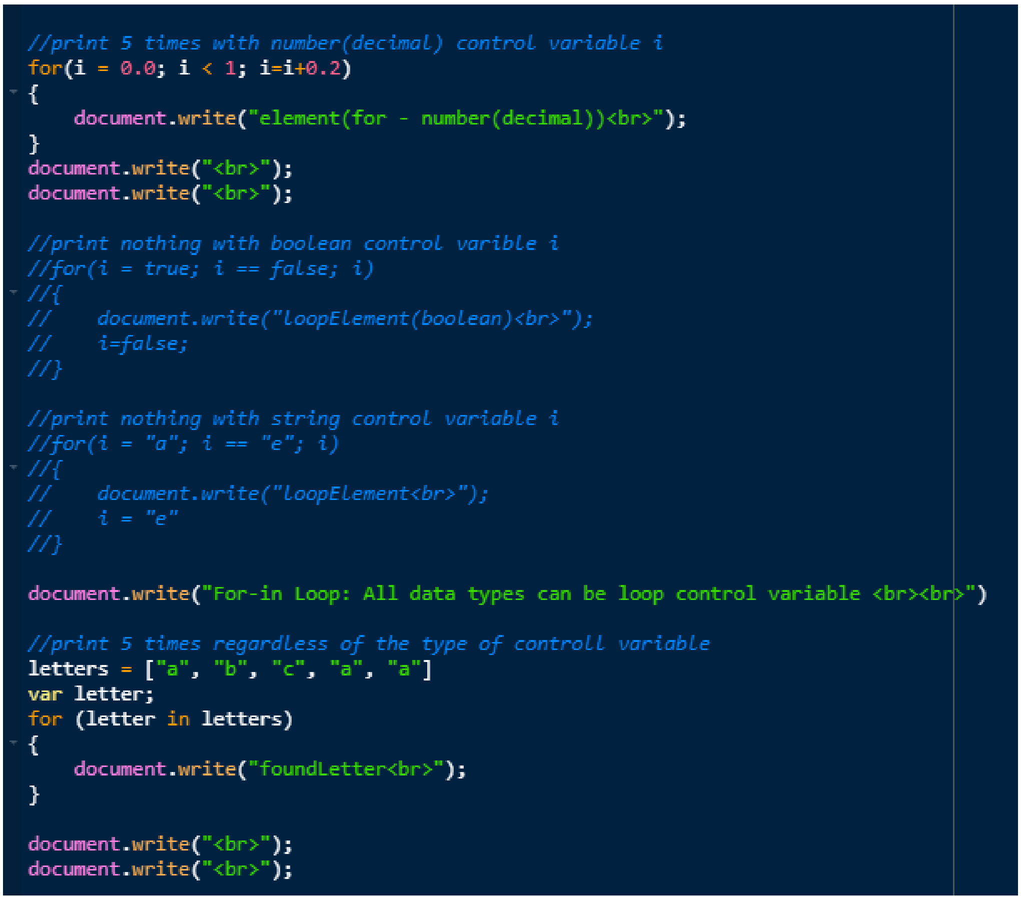
Yes, it is legal for the loop control variable or loop parameters to be changed in the loop. Also change affects loop control.

- **Are the loop parameters evaluated only once, or once for every iteration?**

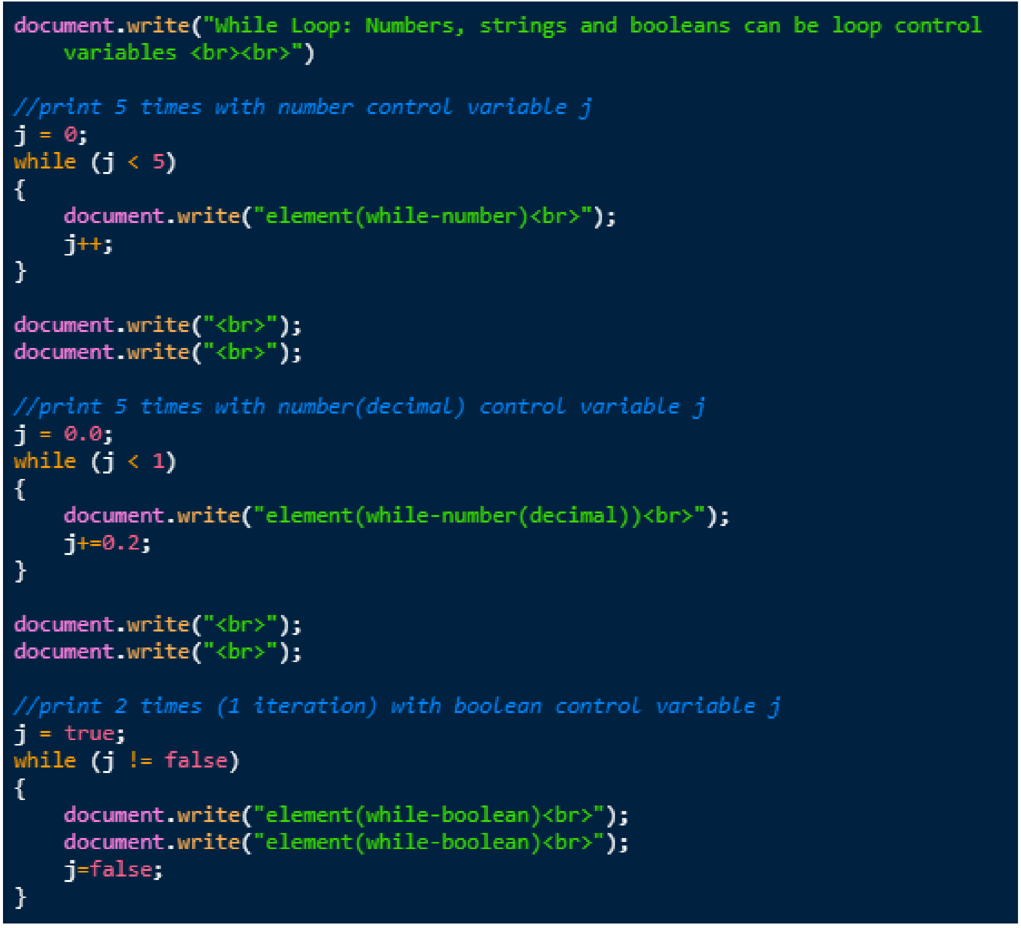
Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have effects on loop variable. So loop parameters are evaluated for every iteration in JavaScript.



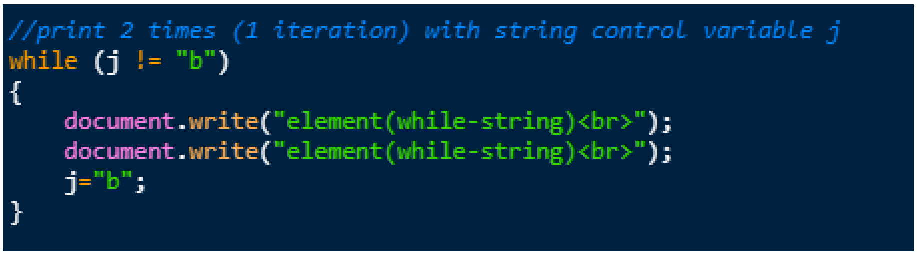
```
//Some lines are commented in sake of execution
document.write("----------Types of Loop Control Variables----------<br><br>");
document.write("For Loop: Numbers can be loop control variable <br><br>")
//print 5 times with number control variable i
for(i = 0; i < 5; i++)
{
    document.write("element(for - number)<br>");
}
document.write("<br>");
document.write("<br>");

//print 5 times with number(decimal) control variable i
for(i = 0.0; i < 1; i=i+0.2)
{
    document.write("element(for - number(decimal))<br>");
}
document.write("<br>");
document.write("<br>");
```



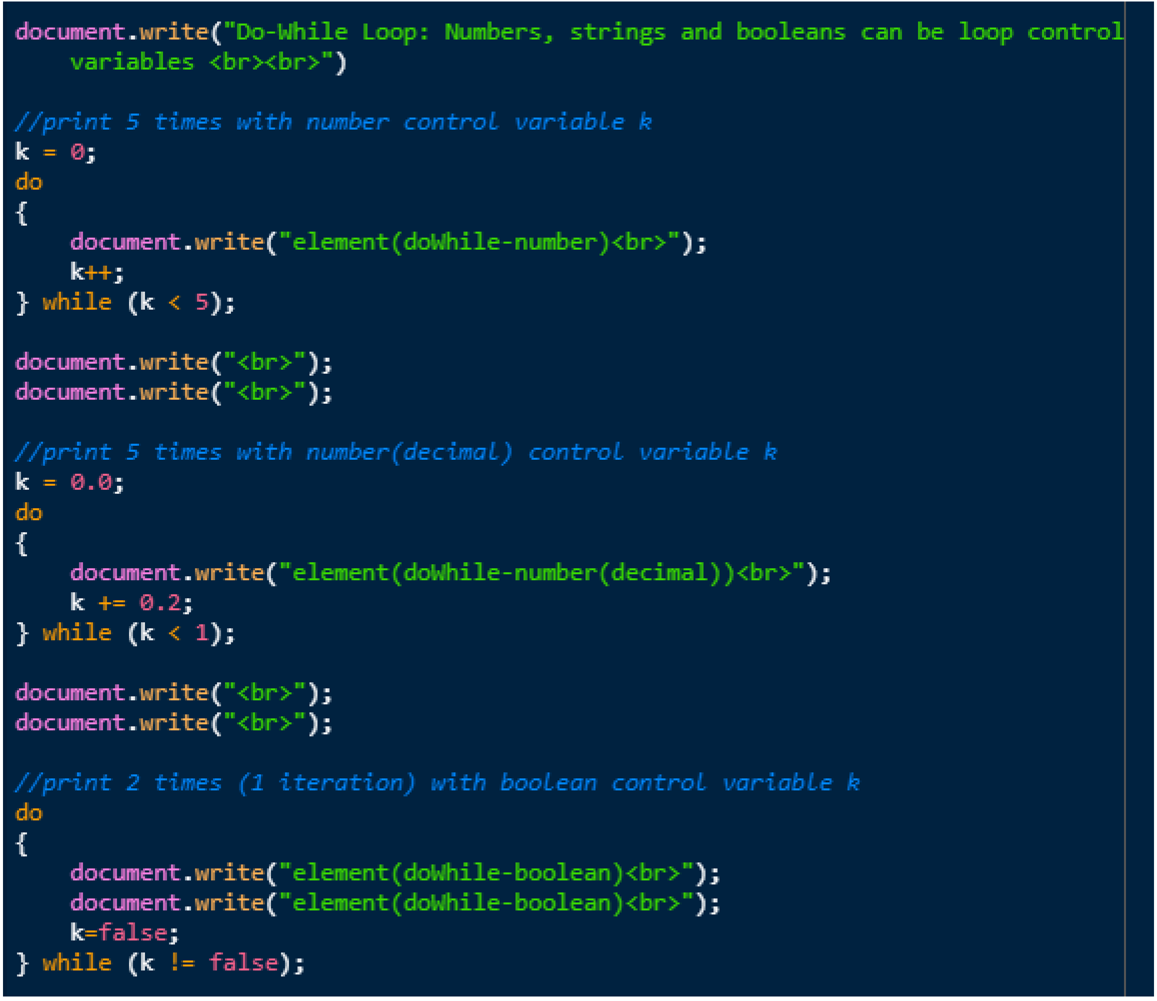
```
//print 5 times with number(decimal) control variable i
for(i = 0.0; i < 1; i=i+0.2)
{
    document.write("element(for - number(decimal))<br>");
}
document.write("<br>");
document.write("<br>");

//print nothing with boolean control varible i
//for(i = true; i == false; i)
//{
//    document.write("loopElement(boolean)<br>");
//    i=false;
//}

//print nothing with string control variable i
//for(i = "a"; i == "e"; i)
//{
//    document.write("loopElement<br>");
//    i = "e"
//}

document.write("For-in Loop: All data types can be loop control variable <br><br>")

//print 5 times regardless of the type of controll variable
letters = ["a", "b", "c", "a", "a"]
var letter;
for (letter in letters)
{
    document.write("foundLetter<br>");
}

document.write("<br>");
document.write("<br>");
```



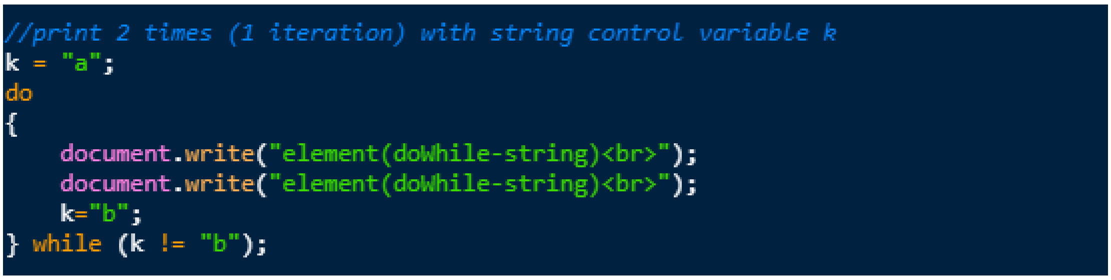
```
document.write("While Loop: Numbers, strings and booleans can be loop control
    variables <br><br>")

//print 5 times with number control variable j
j = 0;
while (j < 5)
{
    document.write("element(while-number)<br>");
    j++;
}

document.write("<br>");
document.write("<br>");

//print 5 times with number(decimal) control variable j
j = 0.0;
while (j < 1)
{
    document.write("element(while-number(decimal))<br>");
    j+=0.2;
}

document.write("<br>");
document.write("<br>");

//print 2 times (1 iteration) with boolean control variable j
j = true;
while (j != false)
{
    document.write("element(while-boolean)<br>");
    document.write("element(while-boolean)<br>");
    j=false;
}
```



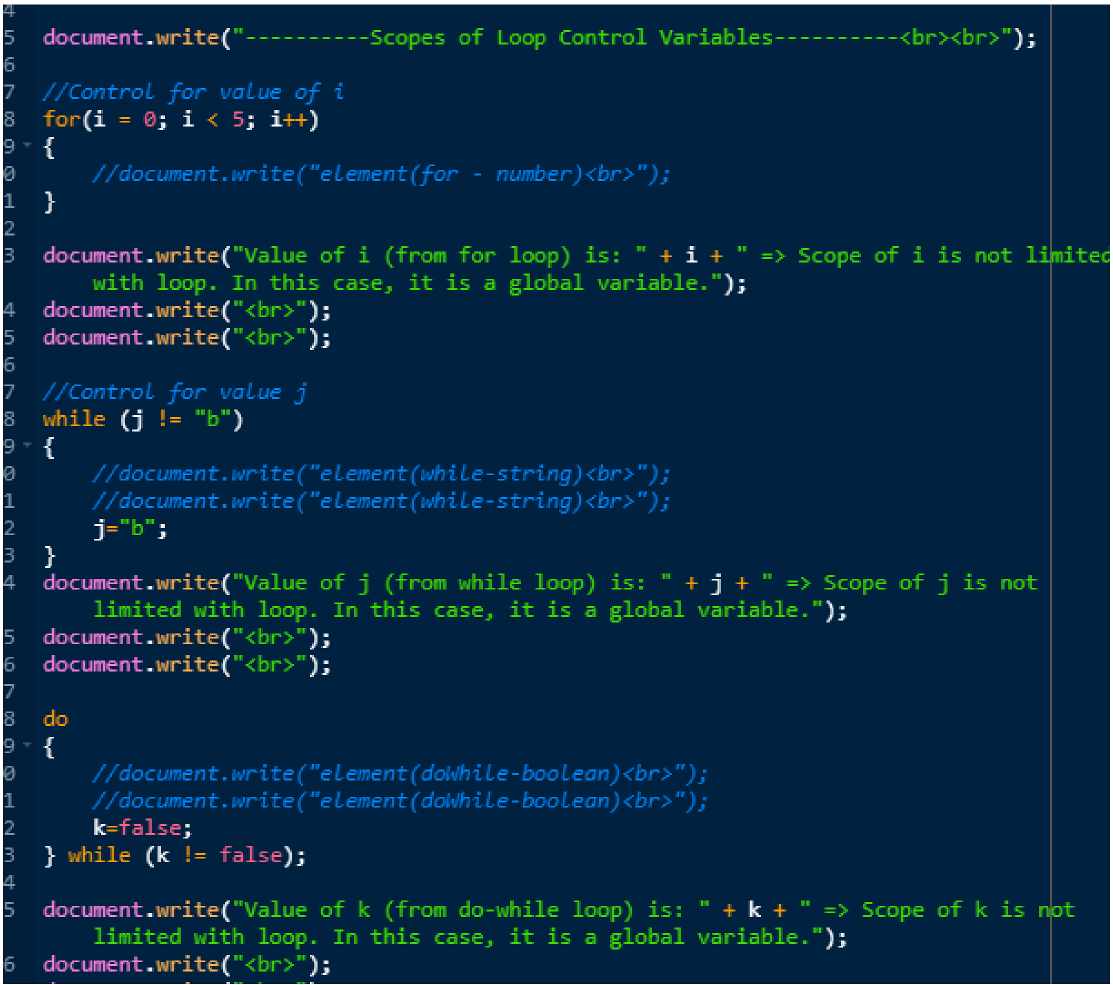
```
//print 2 times (1 iteration) with string control variable j
while (j != "b")
{
    document.write("element(while-string)<br>");
    document.write("element(while-string)<br>");
    j="b";
}
```



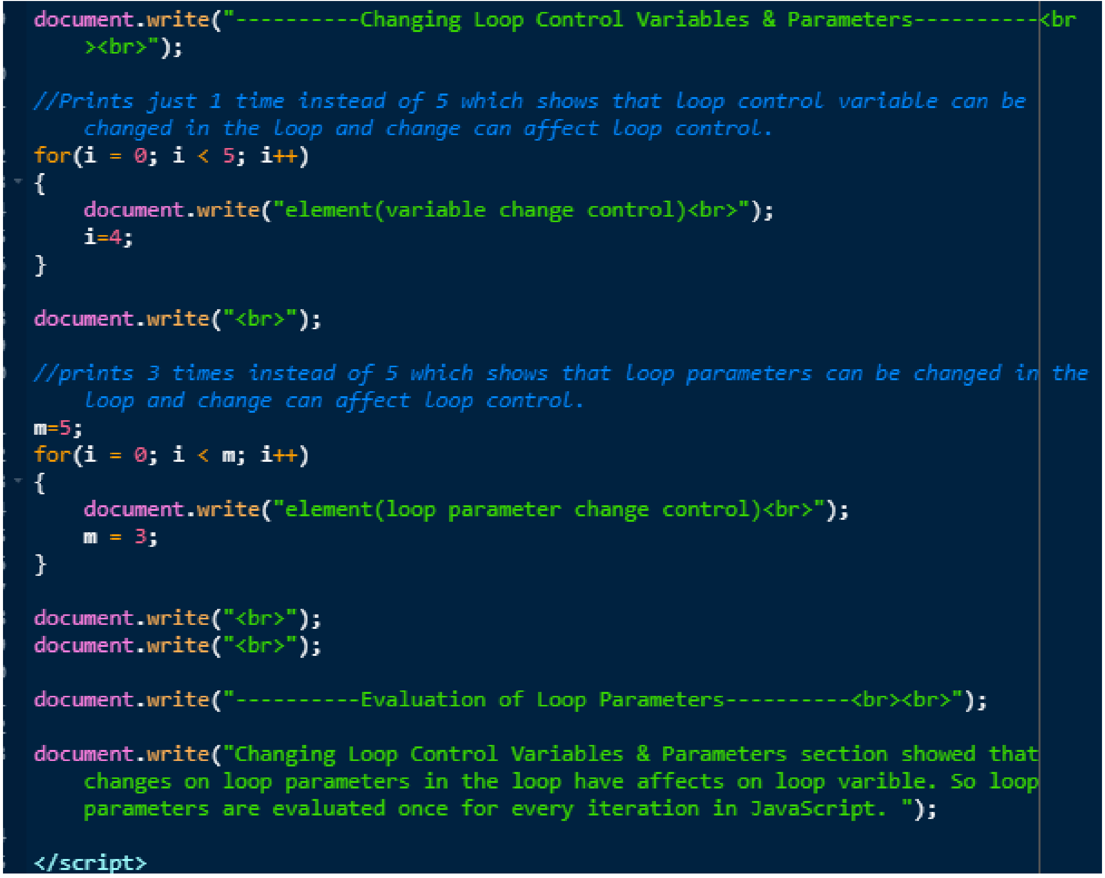
```
document.write("Do-While Loop: Numbers, strings and booleans can be loop control
    variables <br><br>")

//print 5 times with number control variable k
k = 0;
do
{
    document.write("element(doWhile-number)<br>");
    k++;
} while (k < 5);

document.write("<br>");
document.write("<br>");

//print 5 times with number(decimal) control variable k
k = 0.0;
do
{
    document.write("element(doWhile-number(decimal))<br>");
    k += 0.2;
} while (k < 1);

document.write("<br>");
document.write("<br>");

//print 2 times (1 iteration) with boolean control variable k
do
{
    document.write("element(doWhile-boolean)<br>");
    document.write("element(doWhile-boolean)<br>");
    k=false;
} while (k != false);
```



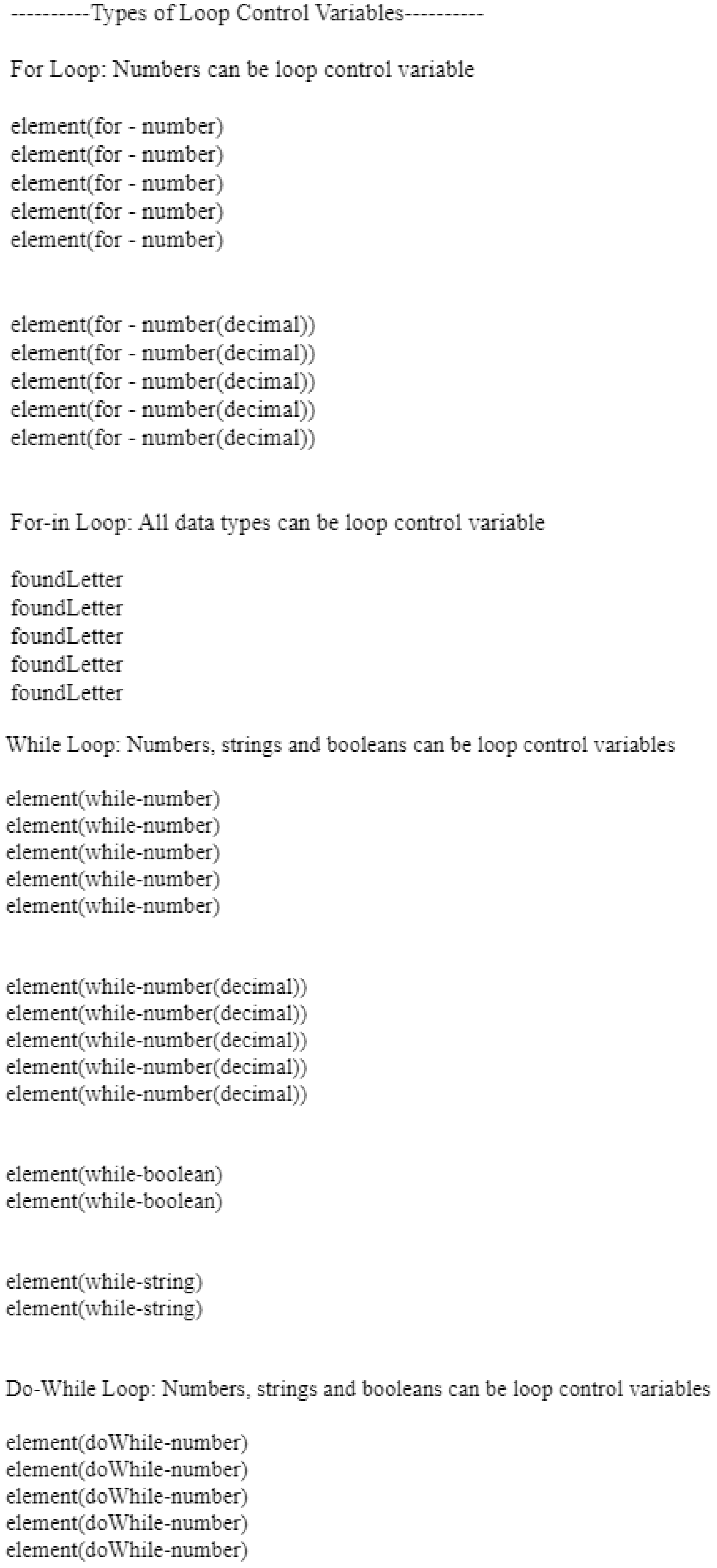
```
//print 2 times (1 iteration) with string control variable k
k = "a";
do
{
    document.write("element(doWhile-string)<br>");
    document.write("element(doWhile-string)<br>");
    k="b";
} while (k != "b");
```



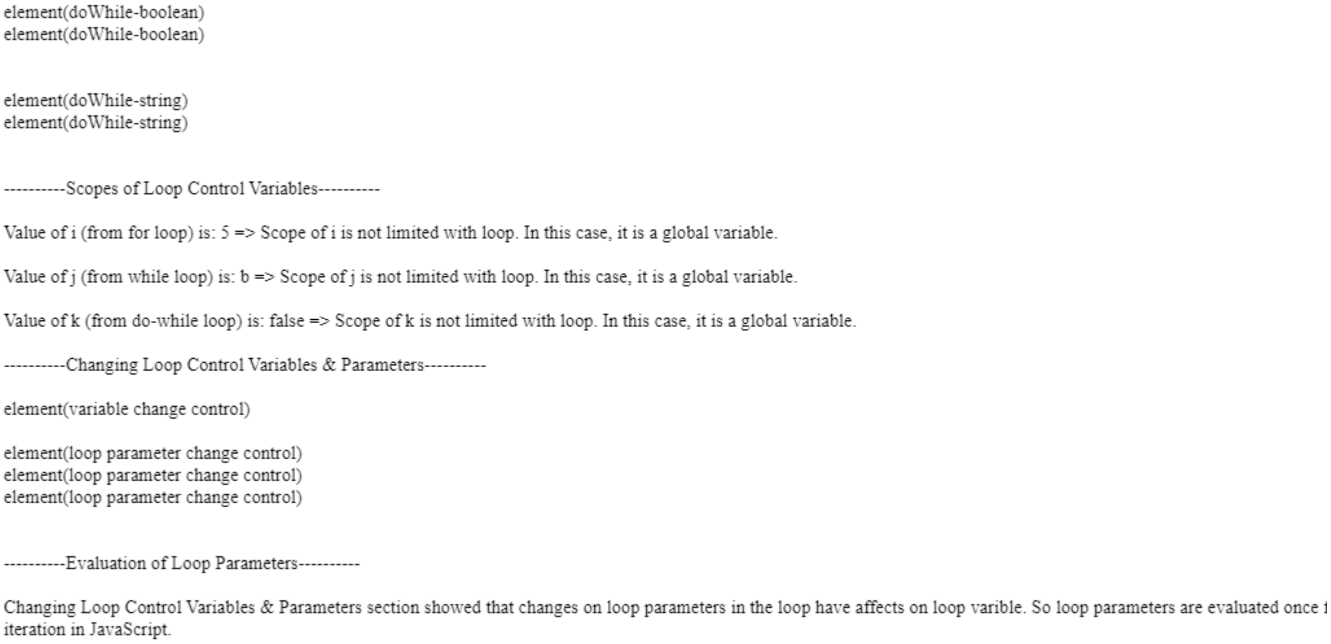
```
document.write("----------Scopes of Loop Control Variables----------<br><br>");

//Control for value of i
for(i = 0; i < 5; i++)
{
    //document.write("element(for - number)<br>");
}

document.write("Value of i (from for loop) is: " + i + " => Scope of i is not limited
    with loop. In this case, it is a global variable.");
document.write("<br>");
document.write("<br>");

//Control for value j
while (j != "b")
{
    //document.write("element(while-string)<br>");
    //document.write("element(while-string)<br>");
    j="b";
}
document.write("Value of j (from while loop) is: " + j + " => Scope of j is not
    limited with loop. In this case, it is a global variable.");
document.write("<br>");
document.write("<br>");

do
{
    //document.write("element(doWhile-boolean)<br>");
    //document.write("element(doWhile-boolean)<br>");
    k=false;
} while (k != false);

document.write("Value of k (from do-while loop) is: " + k + " => Scope of k is not
    limited with loop. In this case, it is a global variable.");
document.write("<br>");
```



```
document.write("----------Changing Loop Control Variables & Parameters----------<br
    ><br>");

//Prints just 1 time instead of 5 which shows that loop control variable can be
    changed in the loop and change can affect loop control.
for(i = 0; i < 5; i++)
{
    document.write("element(variable change control)<br>");
    i=4;
}

document.write("<br>");

//prints 3 times instead of 5 which shows that loop parameters can be changed in the
    loop and change can affect loop control.
m=5;
for(i = 0; i < m; i++)
{
    document.write("element(loop parameter change control)<br>");
    m = 3;
}

document.write("<br>");
document.write("<br>");

document.write("----------Evaluation of Loop Parameters----------<br><br>");

document.write("Changing Loop Control Variables & Parameters section showed that
    changes on loop parameters in the loop have affects on loop varible. So loop
    parameters are evaluated once for every iteration in JavaScript. ");

</script>
```



```
----------Types of Loop Control Variables----------

For Loop: Numbers can be loop control variable

element(for - number)
element(for - number)
element(for - number)
element(for - number)
element(for - number)


element(for - number(decimal))
element(for - number(decimal))
element(for - number(decimal))
element(for - number(decimal))
element(for - number(decimal))


For-in Loop: All data types can be loop control variable

foundLetter
foundLetter
foundLetter
foundLetter
foundLetter

While Loop: Numbers, strings and booleans can be loop control variables

element(while-number)
element(while-number)
element(while-number)
element(while-number)
element(while-number)


element(while-number(decimal))
element(while-number(decimal))
element(while-number(decimal))
element(while-number(decimal))
element(while-number(decimal))


element(while-boolean)
element(while-boolean)


element(while-string)
element(while-string)


Do-While Loop: Numbers, strings and booleans can be loop control variables

element(doWhile-number)
element(doWhile-number)
element(doWhile-number)
element(doWhile-number)
element(doWhile-number)
```



```
element(doWhile-boolean)
element(doWhile-boolean)


element(doWhile-string)
element(doWhile-string)


----------Scopes of Loop Control Variables----------

Value of i (from for loop) is: 5 => Scope of i is not limited with loop. In this case, it is a global variable.

Value of j (from while loop) is: b => Scope of j is not limited with loop. In this case, it is a global variable.

Value of k (from do-while loop) is: false => Scope of k is not limited with loop. In this case, it is a global variable.

----------Changing Loop Control Variables & Parameters----------

element(variable change control)

element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)


----------Evaluation of Loop Parameters----------

Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have affects on loop varible. So loop parameters are evaluated once f
iteration in JavaScript.
```

2. Python

*Details are given in example program.

- **What are the types of loop control variables?**
    - **For:** Integer
    - **While:** Integer, Float, Boolean, String

- **What are the scopes of loop control variables?**

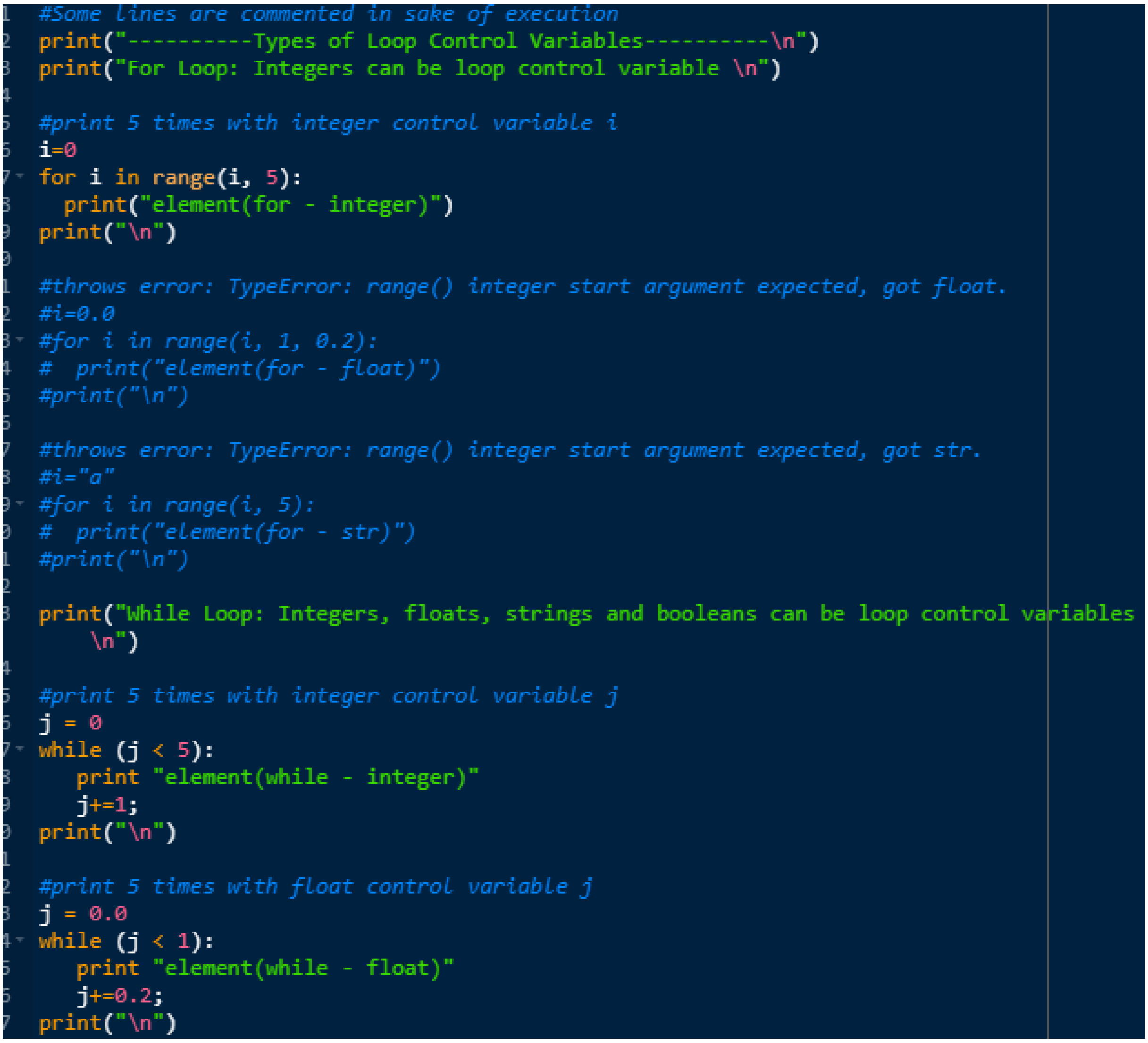
Scope of loop control variables are not limited with the loop and they act like global variable. Loop control variables are visible on all program.

- **Is it legal for the loop control variable or loop parameters to be changed in the loop, and if so, does the change affect loop control?**

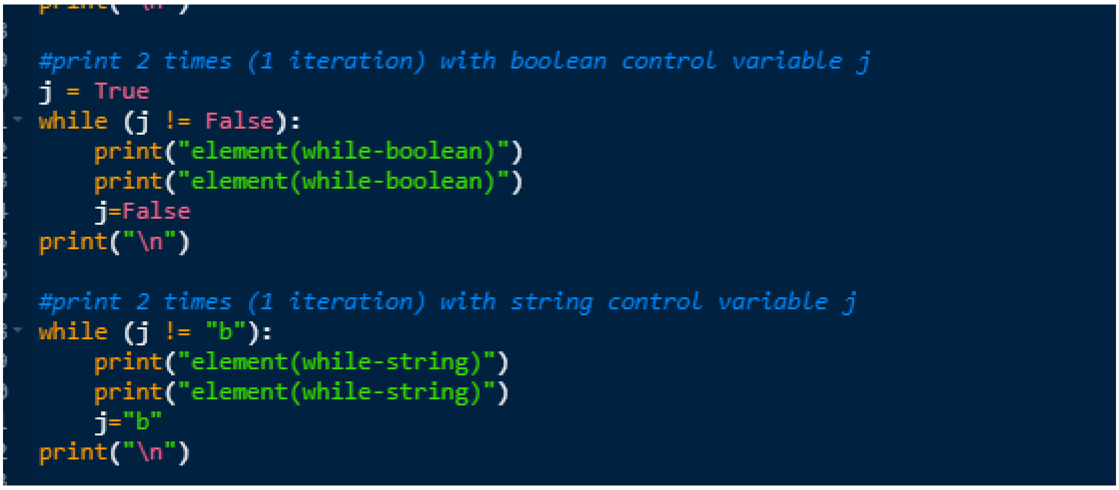
No, it is not legal for the loop control variable or loop parameters to be changed in the loop.

- **Are the loop parameters evaluated only once, or once for every iteration?**

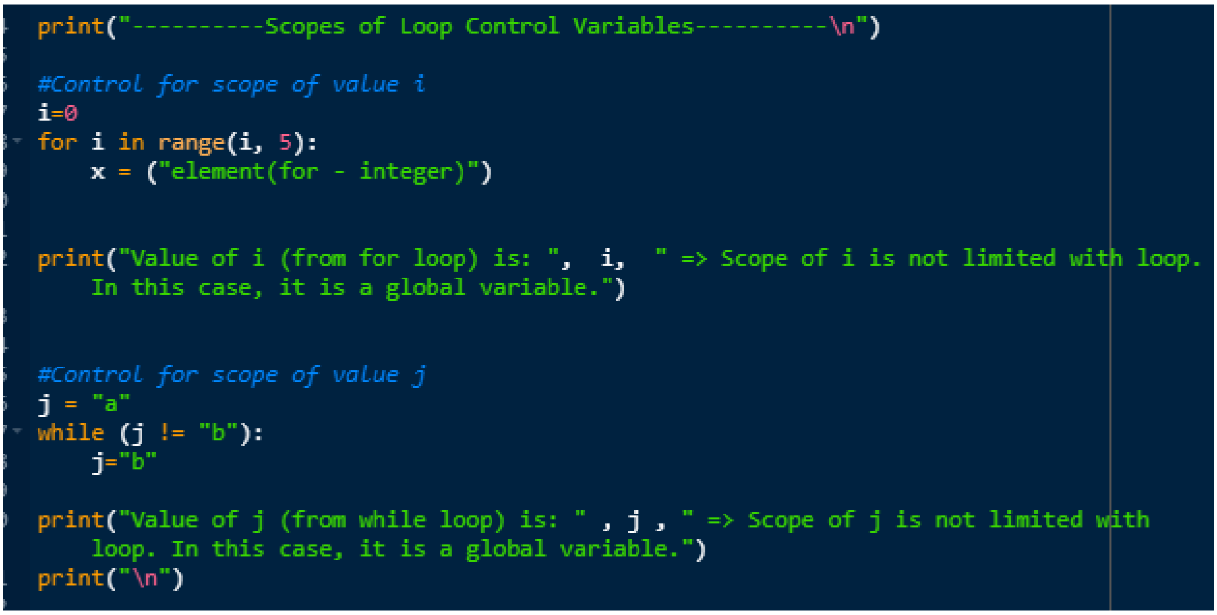
Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop do not have any effect on loop. So loop parameters are evaluated only once before the loop in Python.



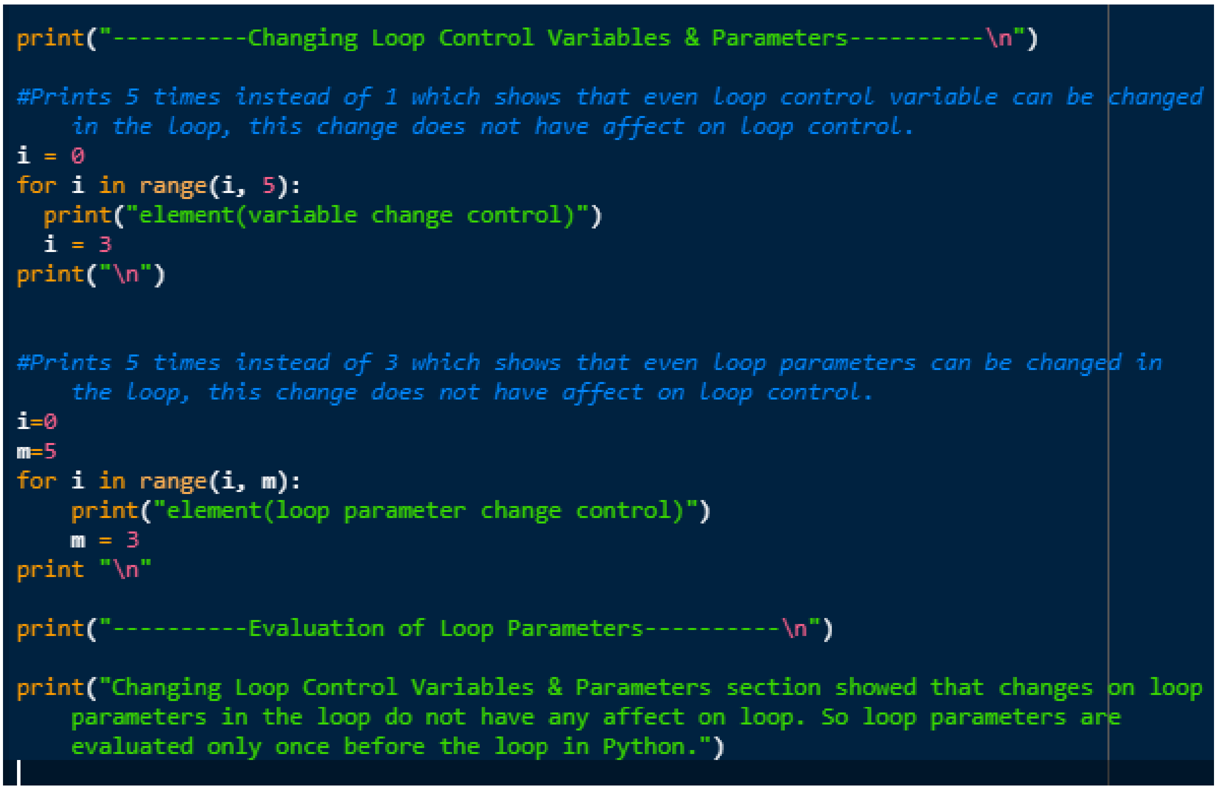
```
#Some lines are commented in sake of execution
print("----------Types of Loop Control Variables----------\n")
print("For Loop: Integers can be loop control variable \n")

#print 5 times with integer control variable i
i=0
for i in range(i, 5):
  print("element(for - integer)")
print("\n")

#throws error: TypeError: range() integer start argument expected, got float.
#i=0.0
#for i in range(i, 1, 0.2):
#  print("element(for - float)")
#print("\n")

#throws error: TypeError: range() integer start argument expected, got str.
#i="a"
#for i in range(i, 5):
#  print("element(for - str)")
#print("\n")

print("While Loop: Integers, floats, strings and booleans can be loop control variables \n")

#print 5 times with integer control variable j
j = 0
while (j < 5):
   print "element(while - integer)"
   j+=1;
print("\n")

#print 5 times with float control variable j
j = 0.0
while (j < 1):
   print "element(while - float)"
   j+=0.2;
print("\n")
```



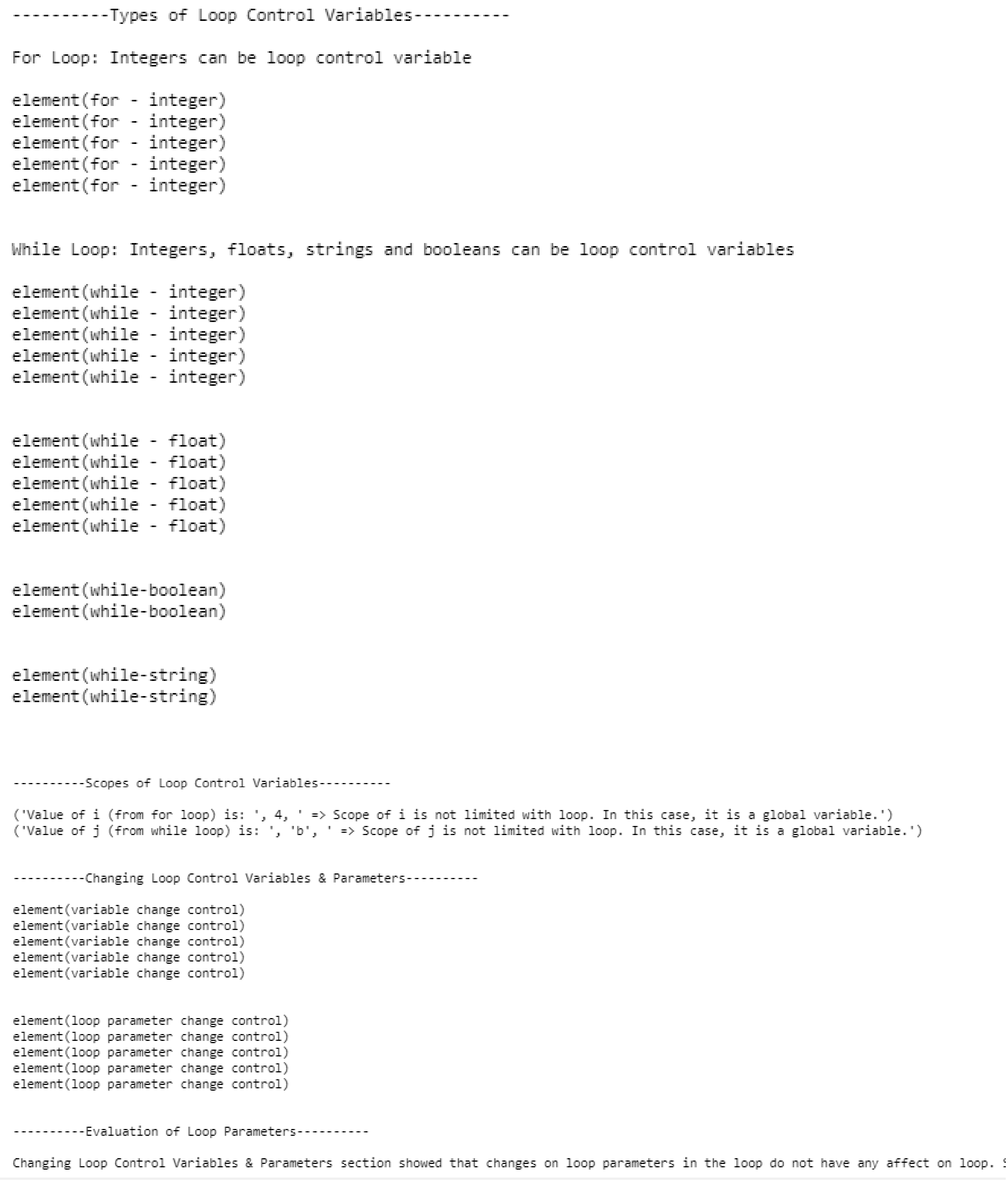
```
print("\n")

#print 2 times (1 iteration) with boolean control variable j
j = True
while (j != False):
    print("element(while-boolean)")
    print("element(while-boolean)")
    j=False
print("\n")

#print 2 times (1 iteration) with string control variable j
while (j != "b"):
    print("element(while-string)")
    print("element(while-string)")
    j="b"
print("\n")
```



```
print("----------Scopes of Loop Control Variables----------\n")

#Control for scope of value i
i=0
for i in range(i, 5):
    x = ("element(for - integer)")


print("Value of i (from for loop) is: ",  i,  " => Scope of i is not limited with loop.
    In this case, it is a global variable.")


#Control for scope of value j
j = "a"
while (j != "b"):
    j="b"

print("Value of j (from while loop) is: " , j , " => Scope of j is not limited with
    loop. In this case, it is a global variable.")
print("\n")
```



```
print("----------Changing Loop Control Variables & Parameters----------\n")

#Prints 5 times instead of 1 which shows that even loop control variable can be changed
    in the loop, this change does not have affect on loop control.
i = 0
for i in range(i, 5):
  print("element(variable change control)")
  i = 3
print("\n")


#Prints 5 times instead of 3 which shows that even loop parameters can be changed in
    the loop, this change does not have affect on loop control.
i=0
m=5
for i in range(i, m):
    print("element(loop parameter change control)")
    m = 3
print "\n"

print("----------Evaluation of Loop Parameters----------\n")

print("Changing Loop Control Variables & Parameters section showed that changes on loop
    parameters in the loop do not have any affect on loop. So loop parameters are
    evaluated only once before the loop in Python.")
```



```
----------Types of Loop Control Variables----------

For Loop: Integers can be loop control variable

element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)


While Loop: Integers, floats, strings and booleans can be loop control variables

element(while - integer)
element(while - integer)
element(while - integer)
element(while - integer)
element(while - integer)


element(while - float)
element(while - float)
element(while - float)
element(while - float)
element(while - float)


element(while-boolean)
element(while-boolean)


element(while-string)
element(while-string)



----------Scopes of Loop Control Variables----------

('Value of i (from for loop) is: ', 4, ' => Scope of i is not limited with loop. In this case, it is a global variable.')
('Value of j (from while loop) is: ', 'b', ' => Scope of j is not limited with loop. In this case, it is a global variable.')


----------Changing Loop Control Variables & Parameters----------

element(variable change control)
element(variable change control)
element(variable change control)
element(variable change control)
element(variable change control)


element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)


----------Evaluation of Loop Parameters----------

Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop do not have any affect on loop.
```

3. PHP

*Details are given in example program.

- **What are the types of loop control variables?**
  - **For:** Integer and Float
  - **For-each:** All data types

- **While:** Integer, Float, String and Boolean
- **Do-while:** Integer, Float, String and Boolean

- **What are the scopes of loop control variables?**

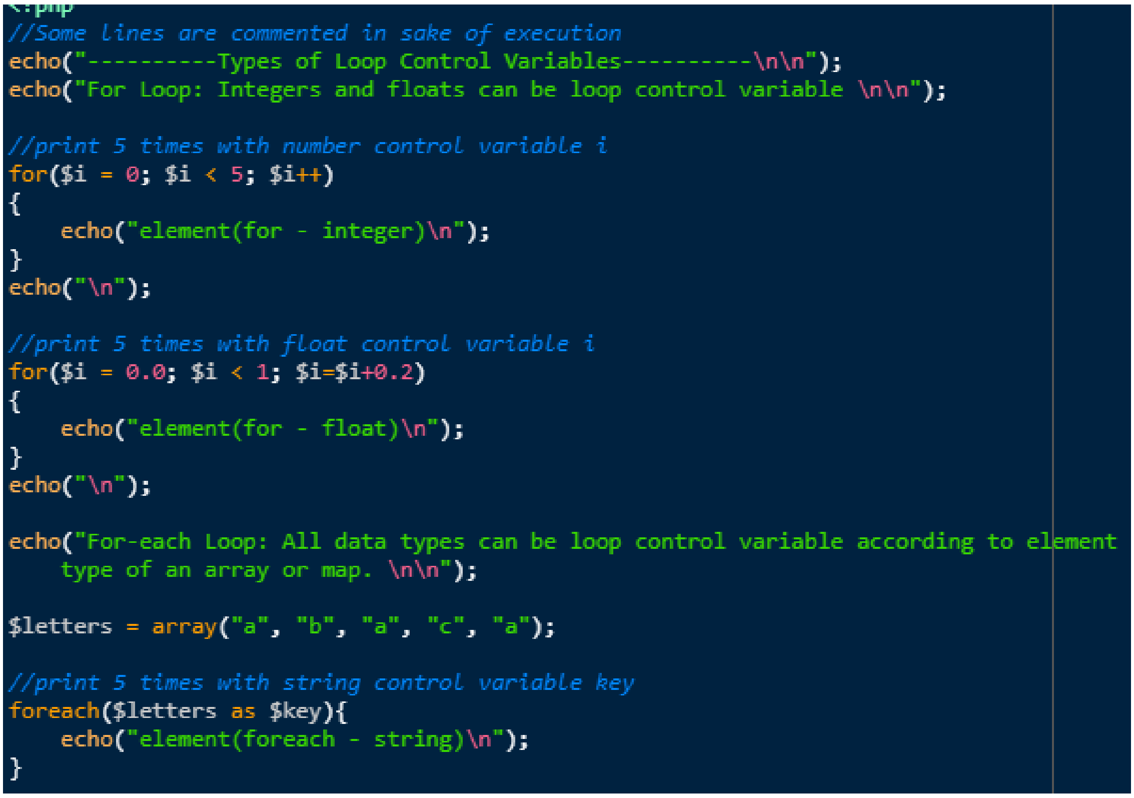
Scope of loop control variables are not limited with the loop and they act like global variable. Loop control variables are visible on all program.

- **Is it legal for the loop control variable or loop parameters to be changed in the loop, and if so, does the change affect loop control?**

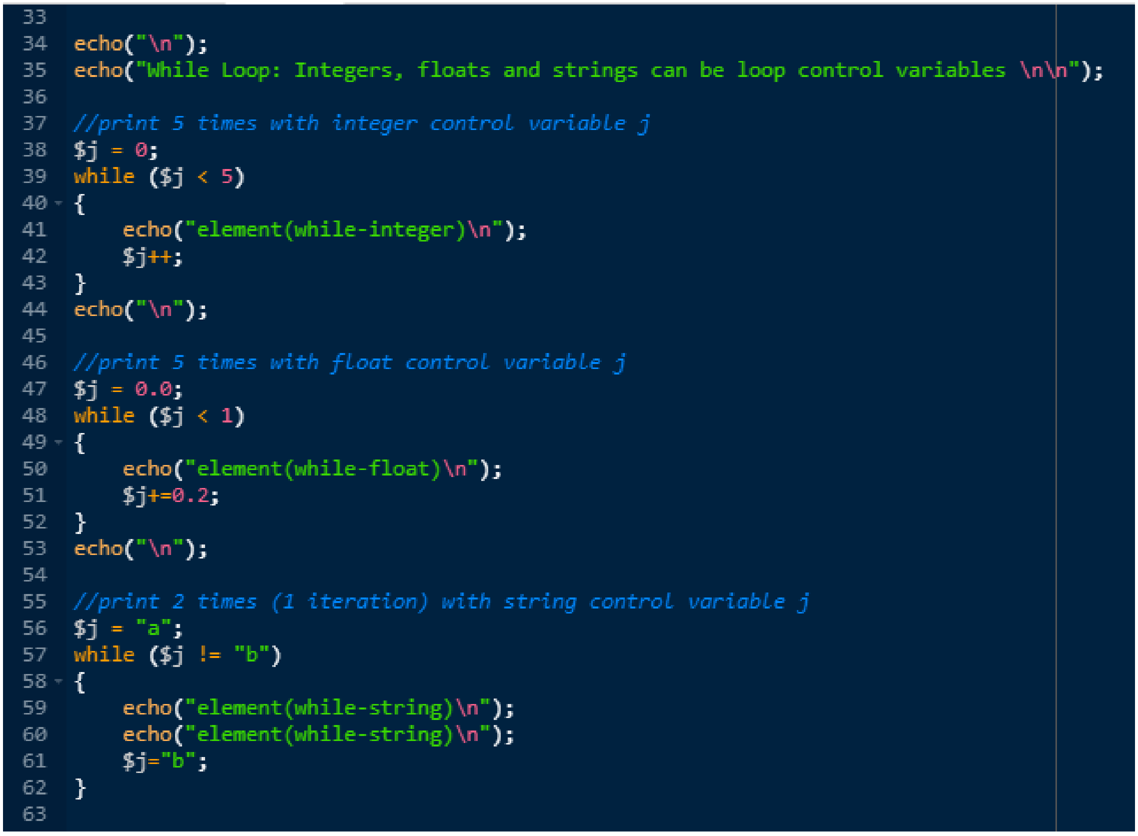
Yes, it is legal for the loop control variable or loop parameters to be changed in the loop. Also change affects loop control.

- **Are the loop parameters evaluated only once, or once for every iteration?**

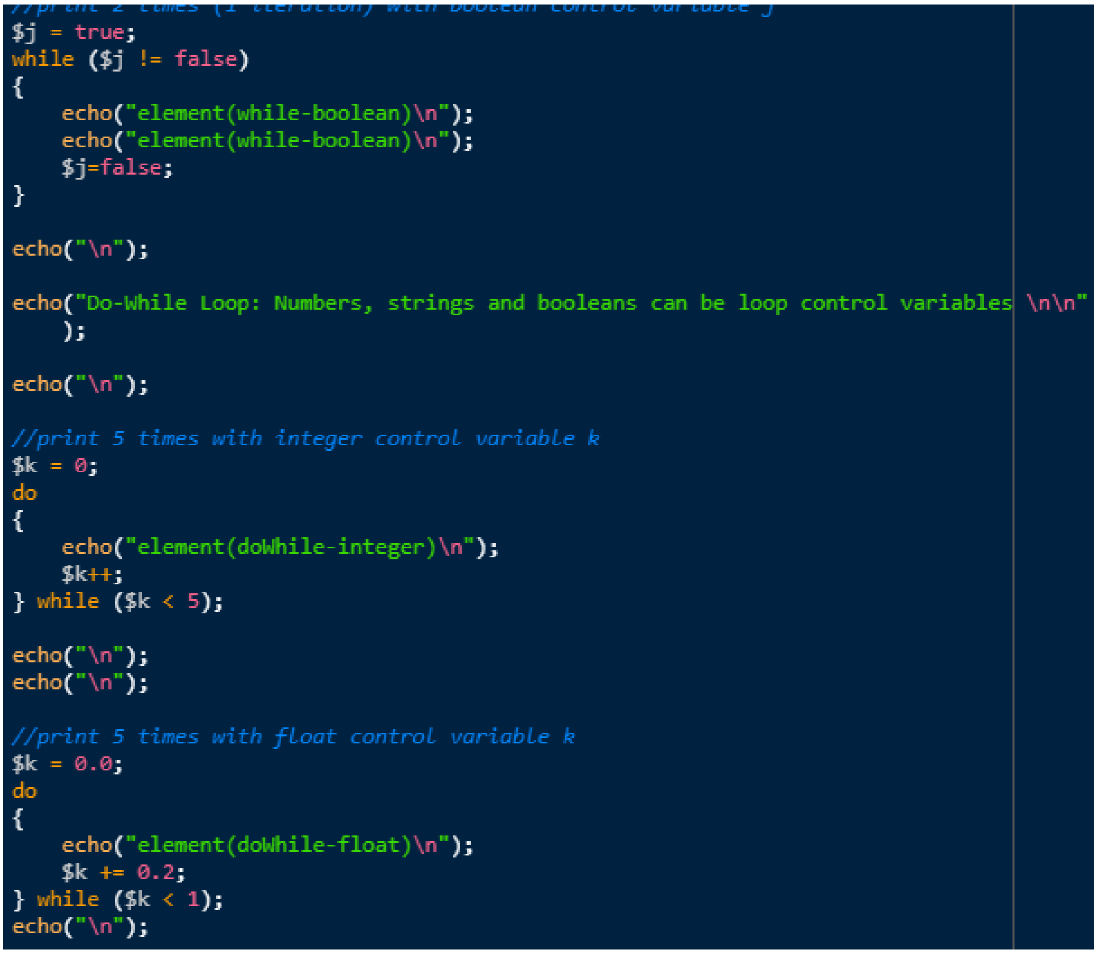
Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have affects on loop variable. So loop parameters are evaluated for every iteration in PHP.



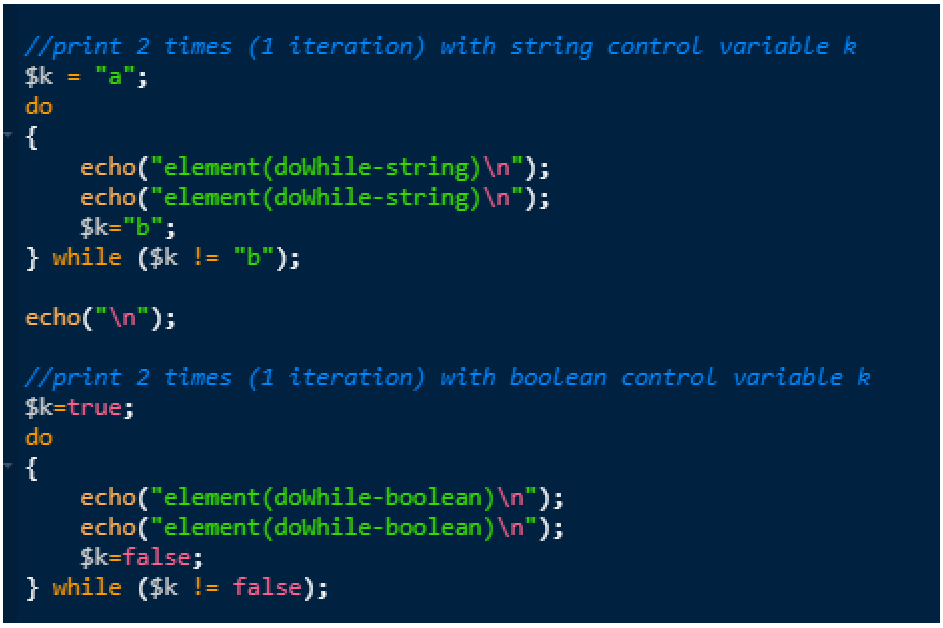
```
<?php
//Some lines are commented in sake of execution
echo("----------Types of Loop Control Variables----------\n\n");
echo("For Loop: Integers and floats can be loop control variable \n\n");

//print 5 times with number control variable i
for($i = 0; $i < 5; $i++)
{
    echo("element(for - integer)\n");
}
echo("\n");

//print 5 times with float control variable i
for($i = 0.0; $i < 1; $i=$i+0.2)
{
    echo("element(for - float)\n");
}
echo("\n");

echo("For-each Loop: All data types can be loop control variable according to element
    type of an array or map. \n\n");

$letters = array("a", "b", "a", "c", "a");

//print 5 times with string control variable key
foreach($letters as $key){
    echo("element(foreach - string)\n");
}
```



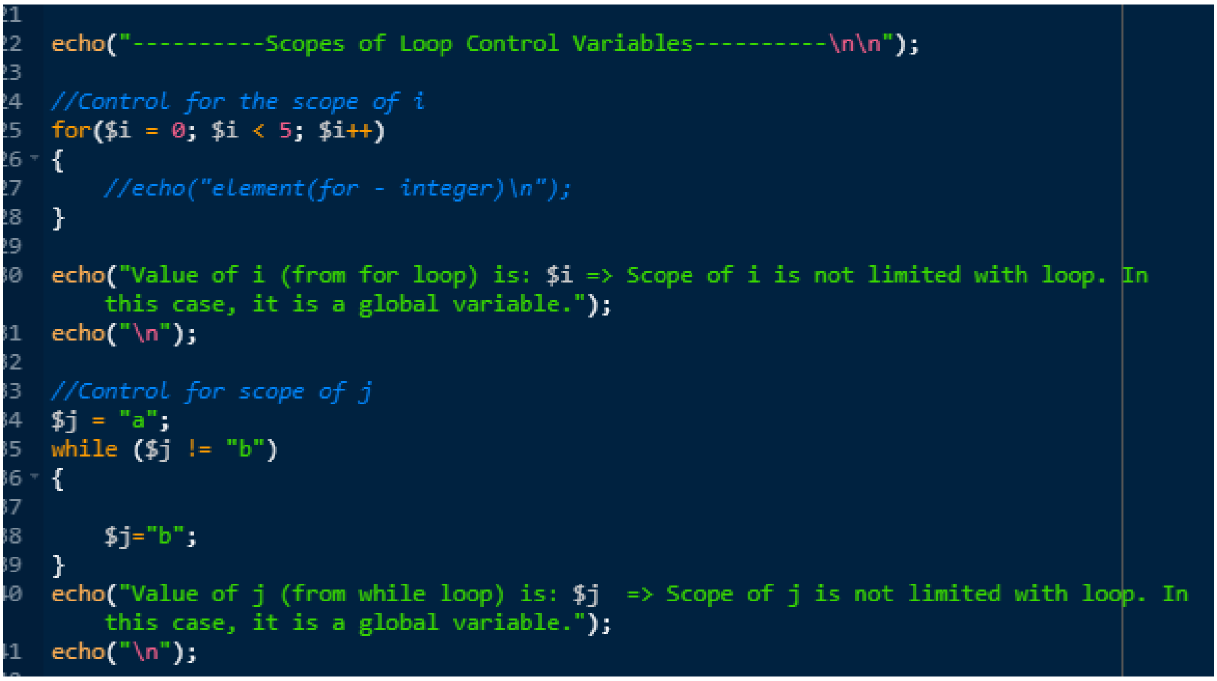
```php

echo("\n");
echo("While Loop: Integers, floats and strings can be loop control variables \n\n");

//print 5 times with integer control variable j
$j = 0;
while ($j < 5)
{
    echo("element(while-integer)\n");
    $j++;
}
echo("\n");

//print 5 times with float control variable j
$j = 0.0;
while ($j < 1)
{
    echo("element(while-float)\n");
    $j+=0.2;
}
echo("\n");

//print 2 times (1 iteration) with string control variable j
$j = "a";
while ($j != "b")
{
    echo("element(while-string)\n");
    echo("element(while-string)\n");
    $j="b";
}

```



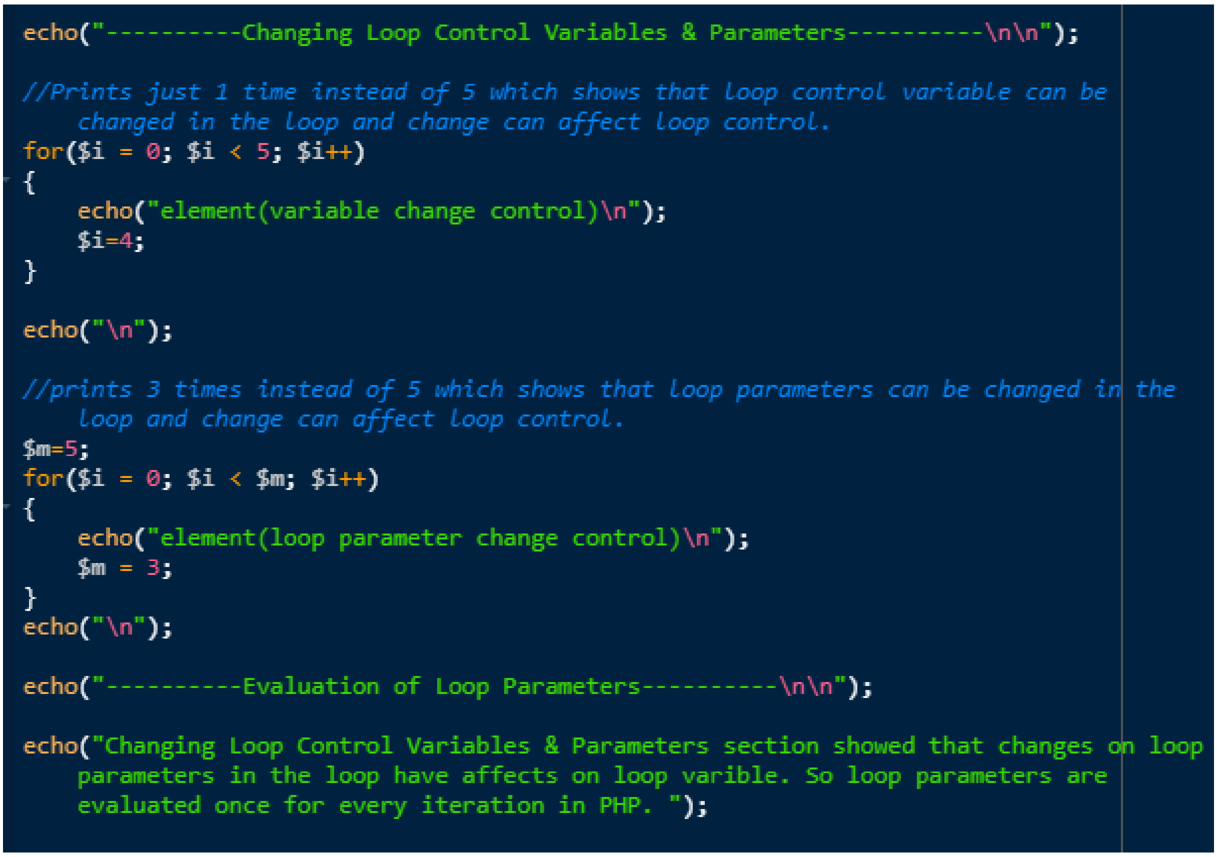
```php
//print 2 times (1 iteration) with boolean control variable j
$j = true;
while ($j != false)
{
    echo("element(while-boolean)\n");
    echo("element(while-boolean)\n");
    $j=false;
}

echo("\n");

echo("Do-While Loop: Numbers, strings and booleans can be loop control variables \n\n"
    );

echo("\n");

//print 5 times with integer control variable k
$k = 0;
do
{
    echo("element(doWhile-integer)\n");
    $k++;
} while ($k < 5);

echo("\n");
echo("\n");

//print 5 times with float control variable k
$k = 0.0;
do
{
    echo("element(doWhile-float)\n");
    $k += 0.2;
} while ($k < 1);
echo("\n");
```



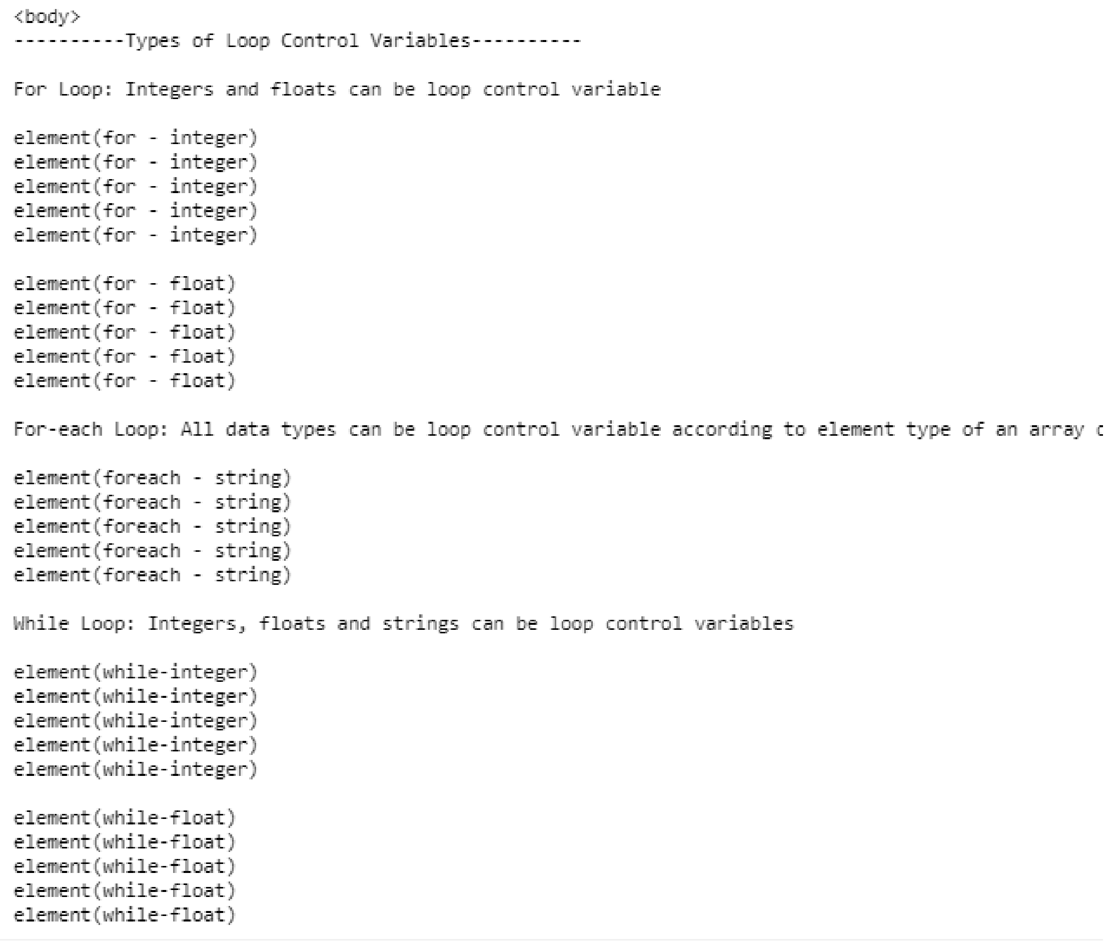
```
//print 2 times (1 iteration) with string control variable k
$k = "a";
do
{
    echo("element(doWhile-string)\n");
    echo("element(doWhile-string)\n");
    $k="b";
} while ($k != "b");

echo("\n");

//print 2 times (1 iteration) with boolean control variable k
$k=true;
do
{
    echo("element(doWhile-boolean)\n");
    echo("element(doWhile-boolean)\n");
    $k=false;
} while ($k != false);
```



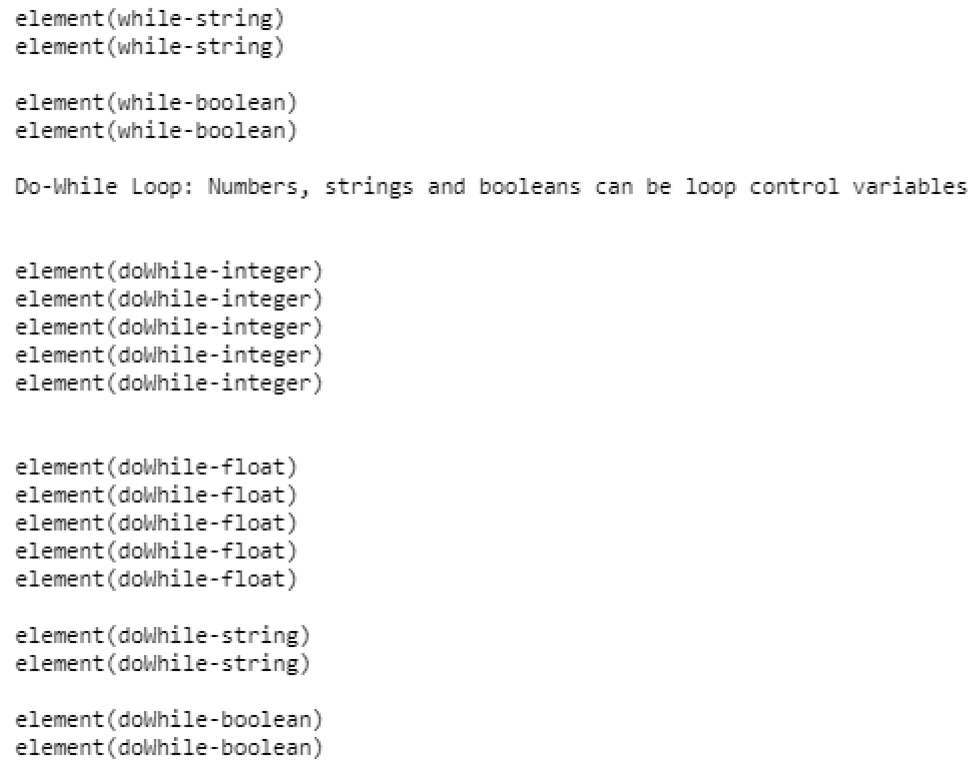
```

echo("----------Scopes of Loop Control Variables----------\n\n");

//Control for the scope of i
for($i = 0; $i < 5; $i++)
{
    //echo("element(for - integer)\n");
}

echo("Value of i (from for loop) is: $i => Scope of i is not limited with loop. In this case, it is a global variable.");
echo("\n");

//Control for scope of j
$j = "a";
while ($j != "b")
{

    $j="b";
}
echo("Value of j (from while loop) is: $j  => Scope of j is not limited with loop. In this case, it is a global variable.");
echo("\n");
```



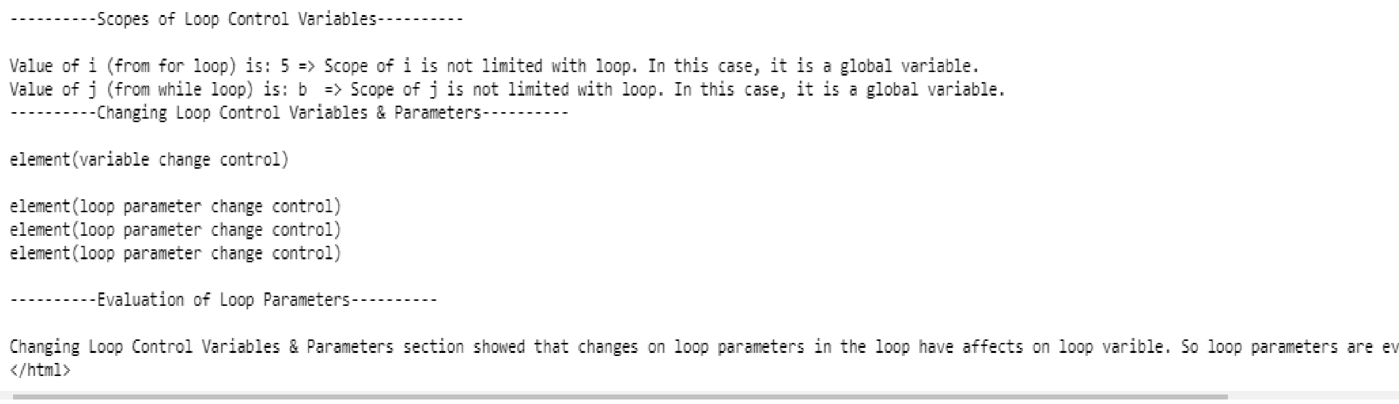
```
echo("----------Changing Loop Control Variables & Parameters----------\n\n");

//Prints just 1 time instead of 5 which shows that loop control variable can be
    changed in the loop and change can affect loop control.
for($i = 0; $i < 5; $i++)
{
    echo("element(variable change control)\n");
    $i=4;
}

echo("\n");

//prints 3 times instead of 5 which shows that loop parameters can be changed in the
    loop and change can affect loop control.
$m=5;
for($i = 0; $i < $m; $i++)
{
    echo("element(loop parameter change control)\n");
    $m = 3;
}
echo("\n");

echo("----------Evaluation of Loop Parameters----------\n\n");

echo("Changing Loop Control Variables & Parameters section showed that changes on loop
    parameters in the loop have affects on loop varible. So loop parameters are
    evaluated once for every iteration in PHP. ");
```



```
<body>
----------Types of Loop Control Variables---------

For Loop: Integers and floats can be loop control variable

element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)

element(for - float)
element(for - float)
element(for - float)
element(for - float)
element(for - float)

For-each Loop: All data types can be loop control variable according to element type of an array c

element(foreach - string)
element(foreach - string)
element(foreach - string)
element(foreach - string)
element(foreach - string)

While Loop: Integers, floats and strings can be loop control variables

element(while-integer)
element(while-integer)
element(while-integer)
element(while-integer)
element(while-integer)

element(while-float)
element(while-float)
element(while-float)
element(while-float)
element(while-float)
```



```
element(while-string)
element(while-string)

element(while-boolean)
element(while-boolean)

Do-While Loop: Numbers, strings and booleans can be loop control variables


element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)


element(doWhile-float)
element(doWhile-float)
element(doWhile-float)
element(doWhile-float)
element(doWhile-float)

element(doWhile-string)
element(doWhile-string)

element(doWhile-boolean)
element(doWhile-boolean)
```



```
----------Scopes of Loop Control Variables----------

Value of i (from for loop) is: 5 => Scope of i is not limited with loop. In this case, it is a global variable.
Value of j (from while loop) is: b  => Scope of j is not limited with loop. In this case, it is a global variable.
----------Changing Loop Control Variables & Parameters----------

element(variable change control)

element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)

----------Evaluation of Loop Parameters----------

Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have affects on loop varible. So loop parameters are ev
</html>
```

4. Dart

*Details are given in example program.

- **What are the types of loop control variables?**

    - **For:** Integer and Double
    - **For-in:** All data types
    - **While:** Integer, Double, String and Boolean
    - **Do-while:** Integer, Float, String and Boolean

- **What are the scopes of loop control variables?**

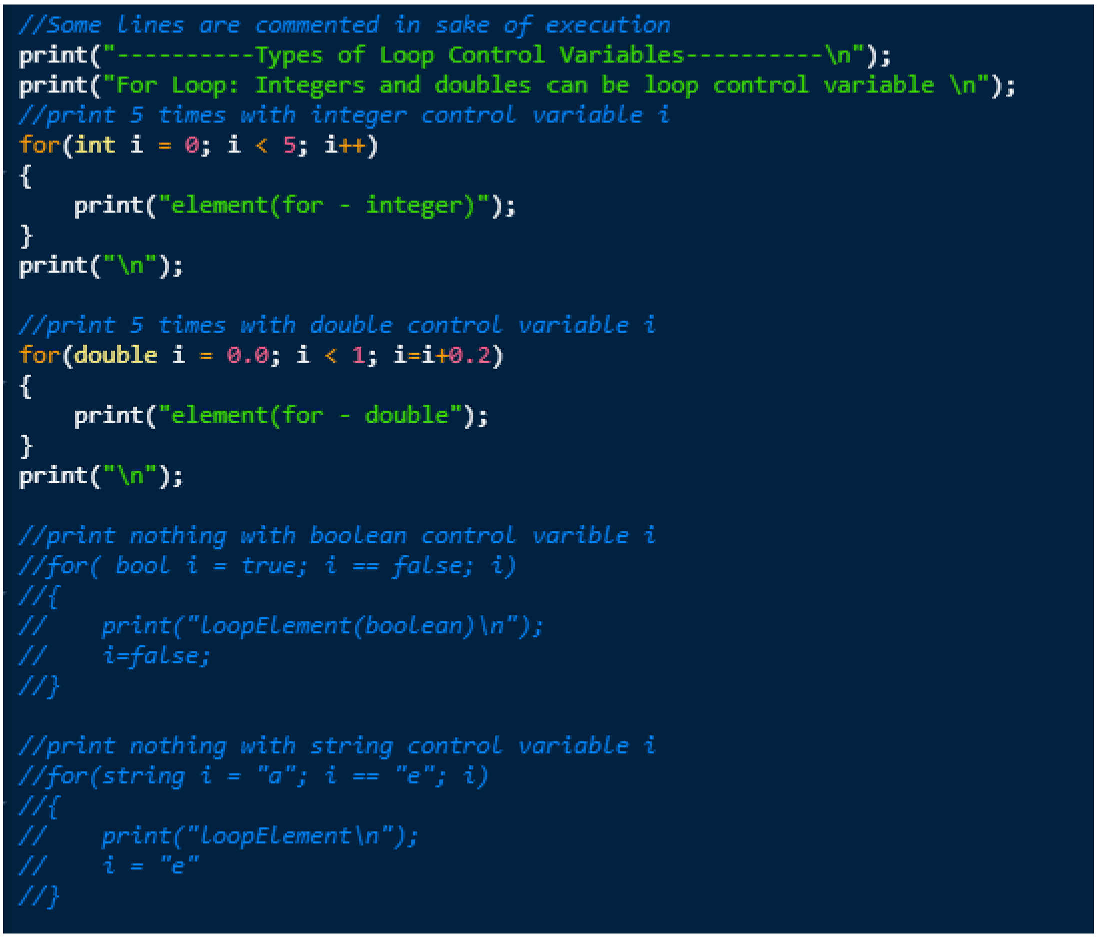
Scope of loop control variables of for loop is limited with the loop and it is not visible in global scope.

- **Is it legal for the loop control variable or loop parameters to be changed in the loop, and if so, does the change affect loop control?**

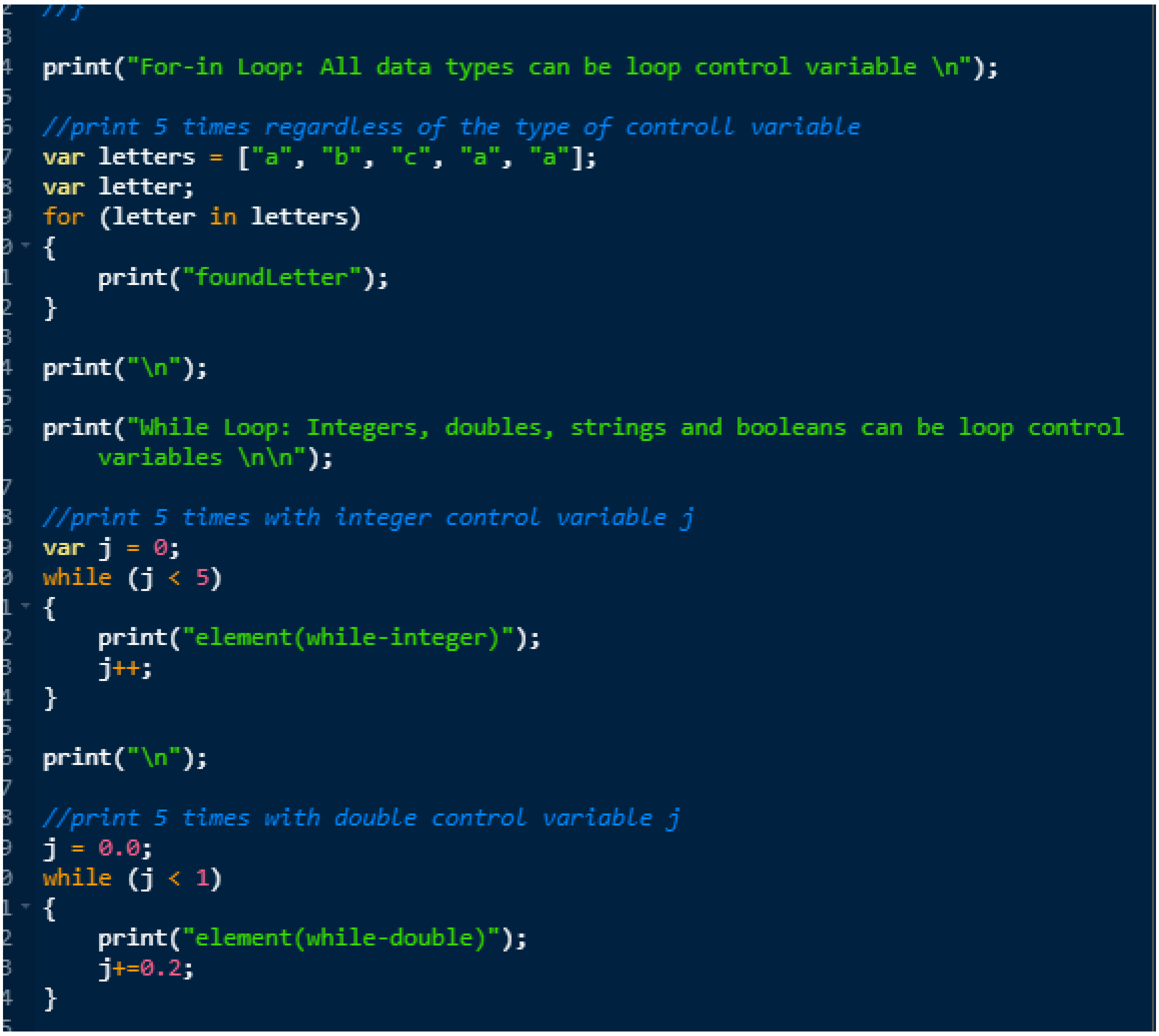
Yes, it is legal for the loop control variable or loop parameters to be changed in the loop. Also change affects loop control.

- **Are the loop parameters evaluated only once, or once for every iteration?**

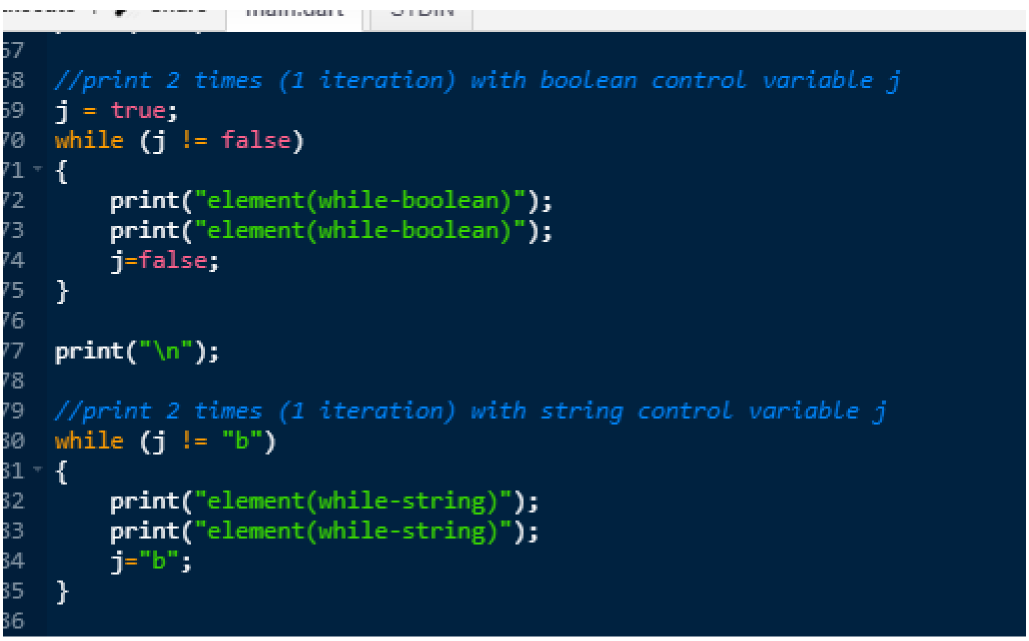
Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have affects on loop variable. So loop parameters are evaluated for every iteration in Dart.



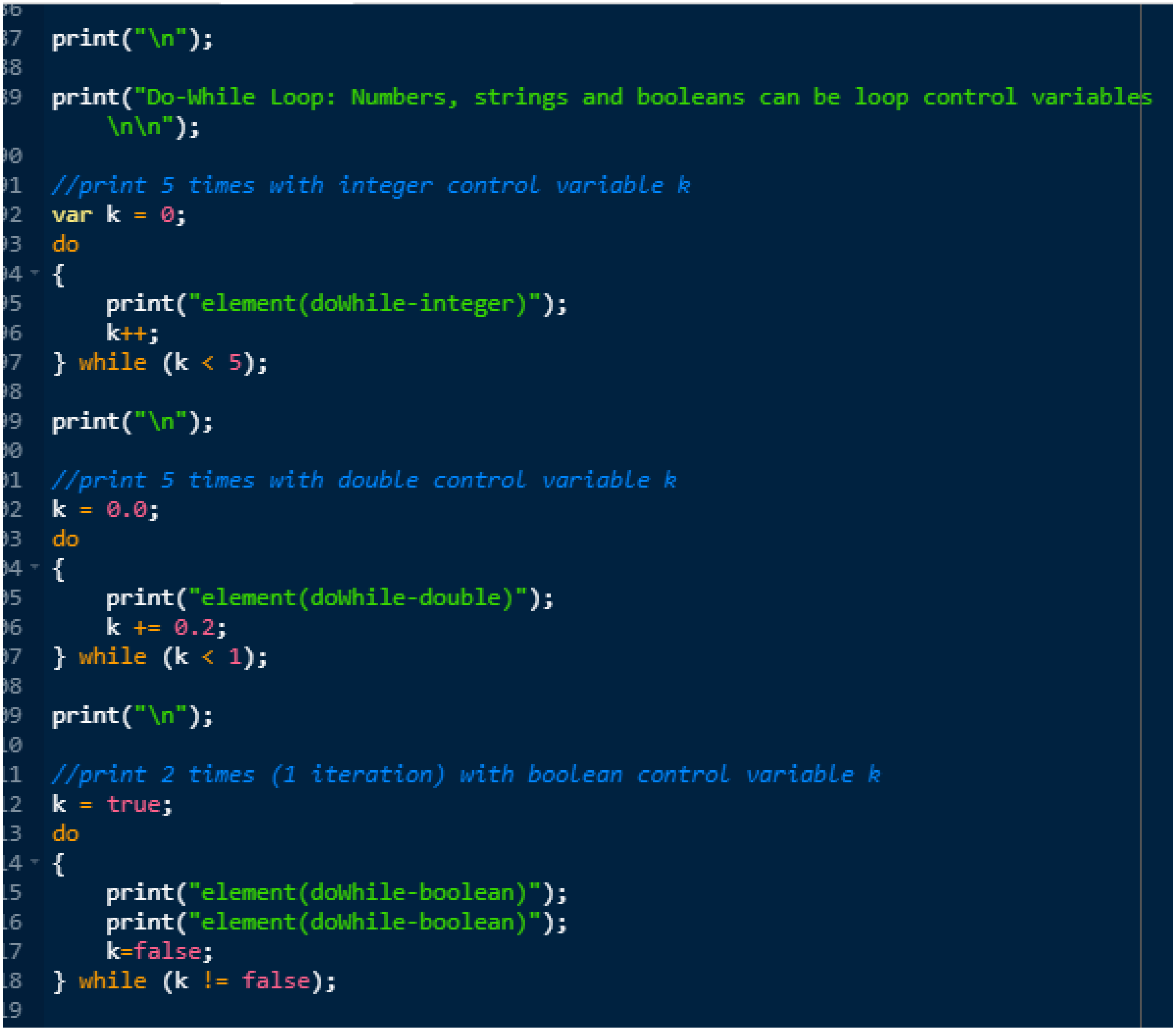
```
//Some lines are commented in sake of execution
print("----------Types of Loop Control Variables----------\n");
print("For Loop: Integers and doubles can be loop control variable \n");
//print 5 times with integer control variable i
for(int i = 0; i < 5; i++)
{
    print("element(for - integer)");
}
print("\n");

//print 5 times with double control variable i
for(double i = 0.0; i < 1; i=i+0.2)
{
    print("element(for - double");
}
print("\n");

//print nothing with boolean control varible i
//for( bool i = true; i == false; i)
//{
//    print("loopElement(boolean)\n");
//    i=false;
//}

//print nothing with string control variable i
//for(string i = "a"; i == "e"; i)
//{
//    print("loopElement\n");
//    i = "e"
//}
```



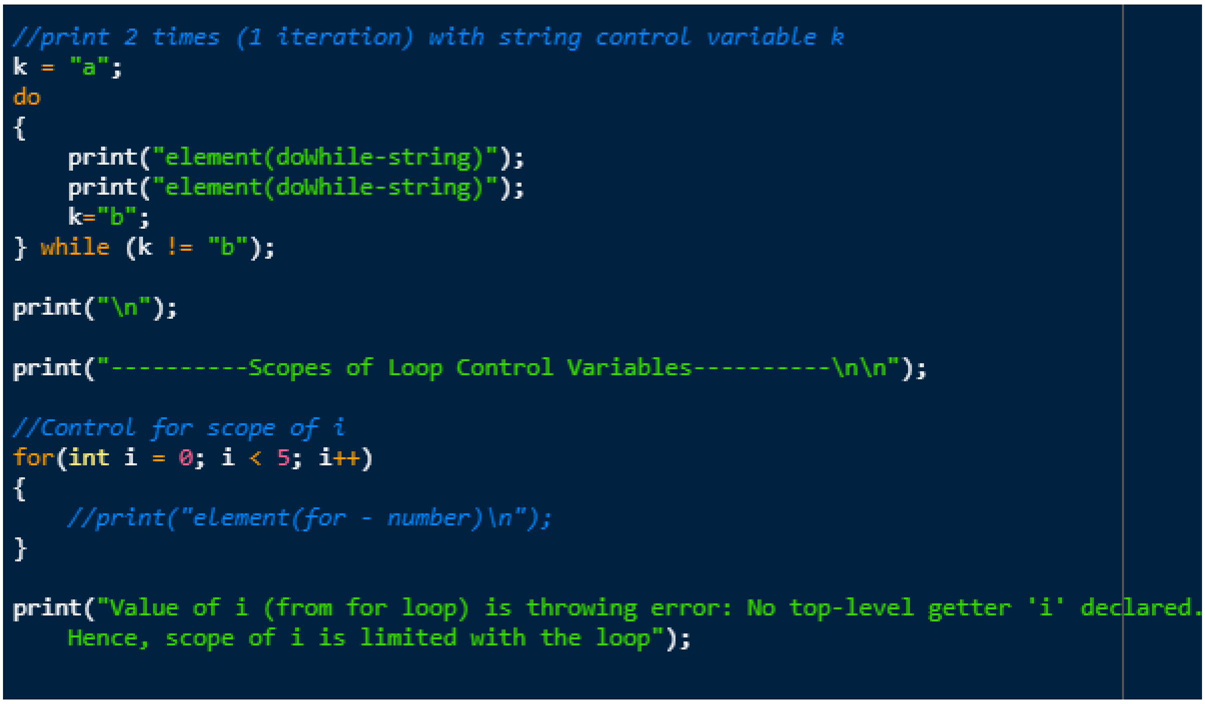
```
//}

print("For-in Loop: All data types can be loop control variable \n");

//print 5 times regardless of the type of controll variable
var letters = ["a", "b", "c", "a", "a"];
var letter;
for (letter in letters)
{
    print("foundLetter");
}

print("\n");

print("While Loop: Integers, doubles, strings and booleans can be loop control
    variables \n\n");

//print 5 times with integer control variable j
var j = 0;
while (j < 5)
{
    print("element(while-integer)");
    j++;
}

print("\n");

//print 5 times with double control variable j
j = 0.0;
while (j < 1)
{
    print("element(while-double)");
    j+=0.2;
}
```



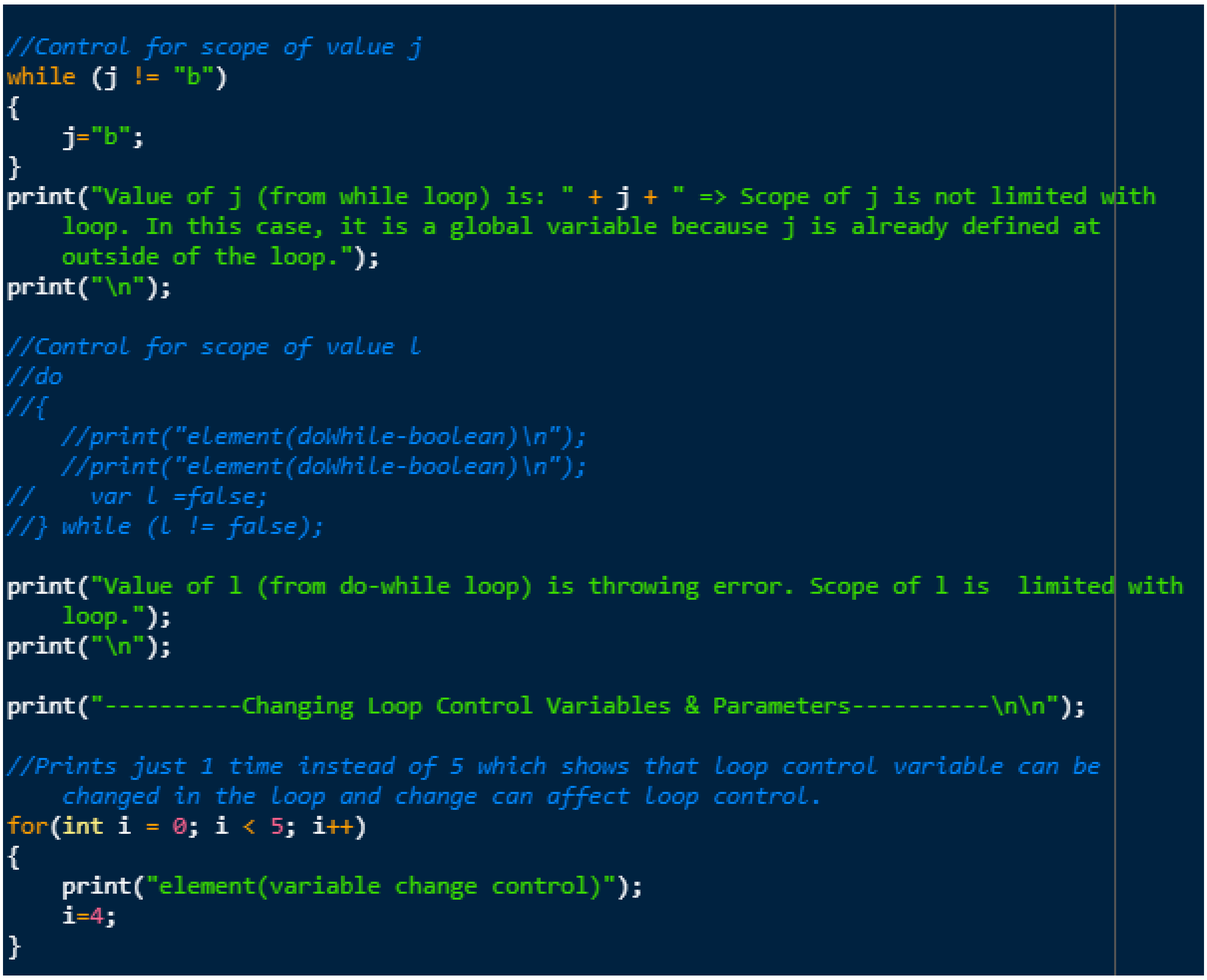
```

//print 2 times (1 iteration) with boolean control variable j
j = true;
while (j != false)
{
    print("element(while-boolean)");
    print("element(while-boolean)");
    j=false;
}

print("\n");

//print 2 times (1 iteration) with string control variable j
while (j != "b")
{
    print("element(while-string)");
    print("element(while-string)");
    j="b";
}

```



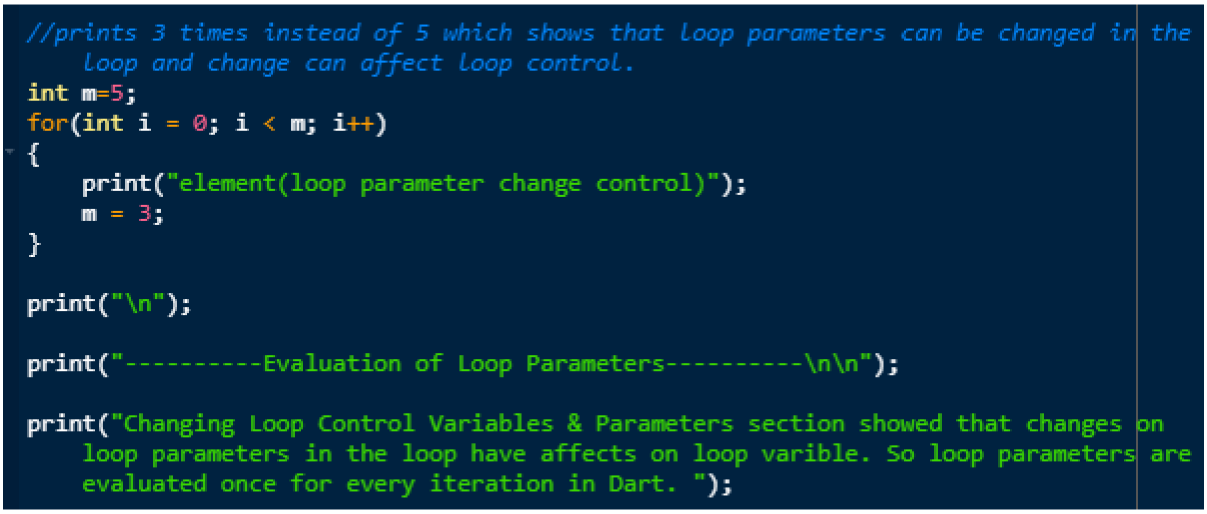
```
print("\n");

print("Do-While Loop: Numbers, strings and booleans can be loop control variables
    \n\n");

//print 5 times with integer control variable k
var k = 0;
do
{
    print("element(doWhile-integer)");
    k++;
} while (k < 5);

print("\n");

//print 5 times with double control variable k
k = 0.0;
do
{
    print("element(doWhile-double)");
    k += 0.2;
} while (k < 1);

print("\n");

//print 2 times (1 iteration) with boolean control variable k
k = true;
do
{
    print("element(doWhile-boolean)");
    print("element(doWhile-boolean)");
    k=false;
} while (k != false);
```



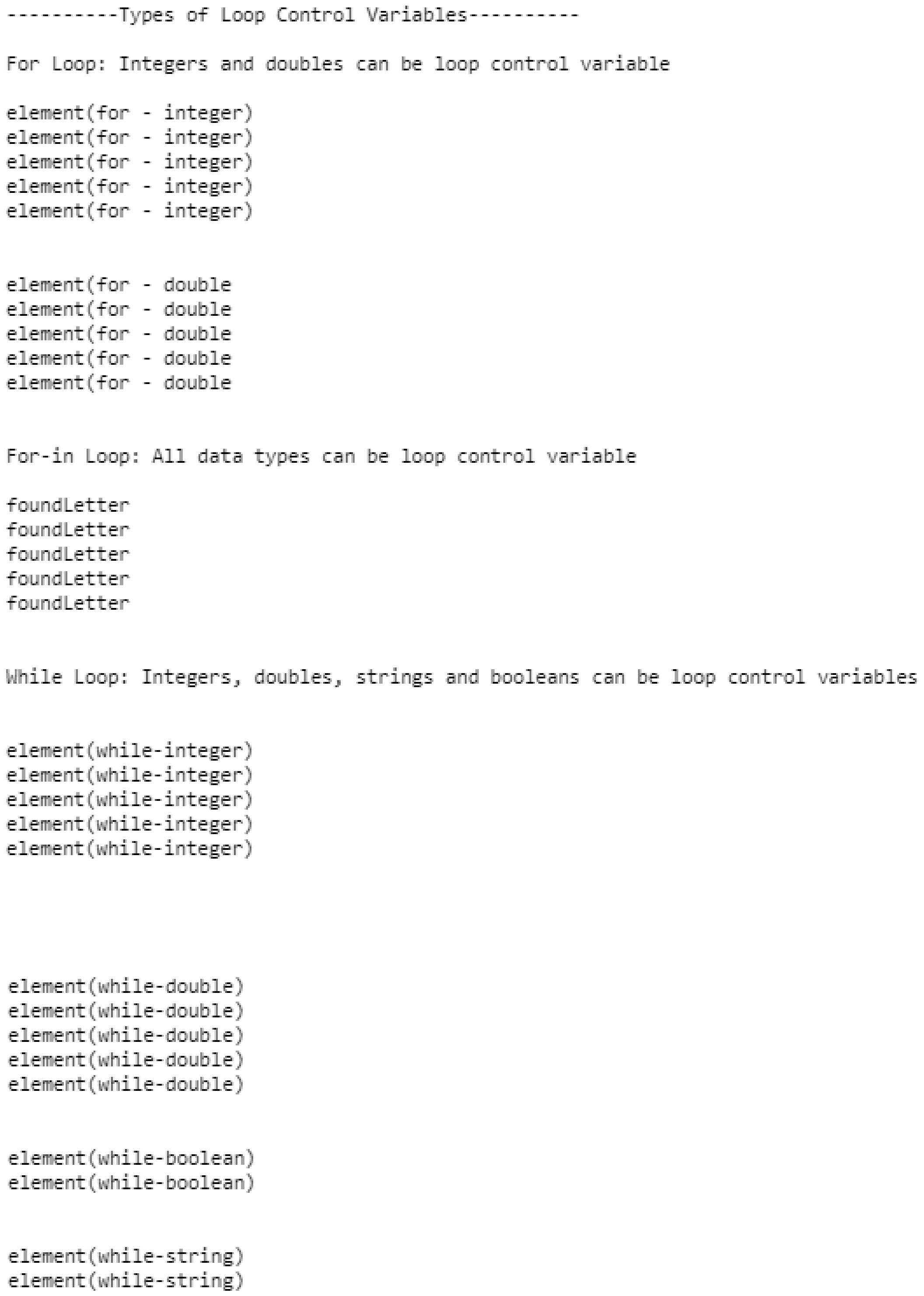
```
//print 2 times (1 iteration) with string control variable k
k = "a";
do
{
    print("element(doWhile-string)");
    print("element(doWhile-string)");
    k="b";
} while (k != "b");

print("\n");

print("----------Scopes of Loop Control Variables----------\n\n");

//Control for scope of i
for(int i = 0; i < 5; i++)
{
    //print("element(for - number)\n");
}

print("Value of i (from for loop) is throwing error: No top-level getter 'i' declared.
    Hence, scope of i is limited with the loop");
```



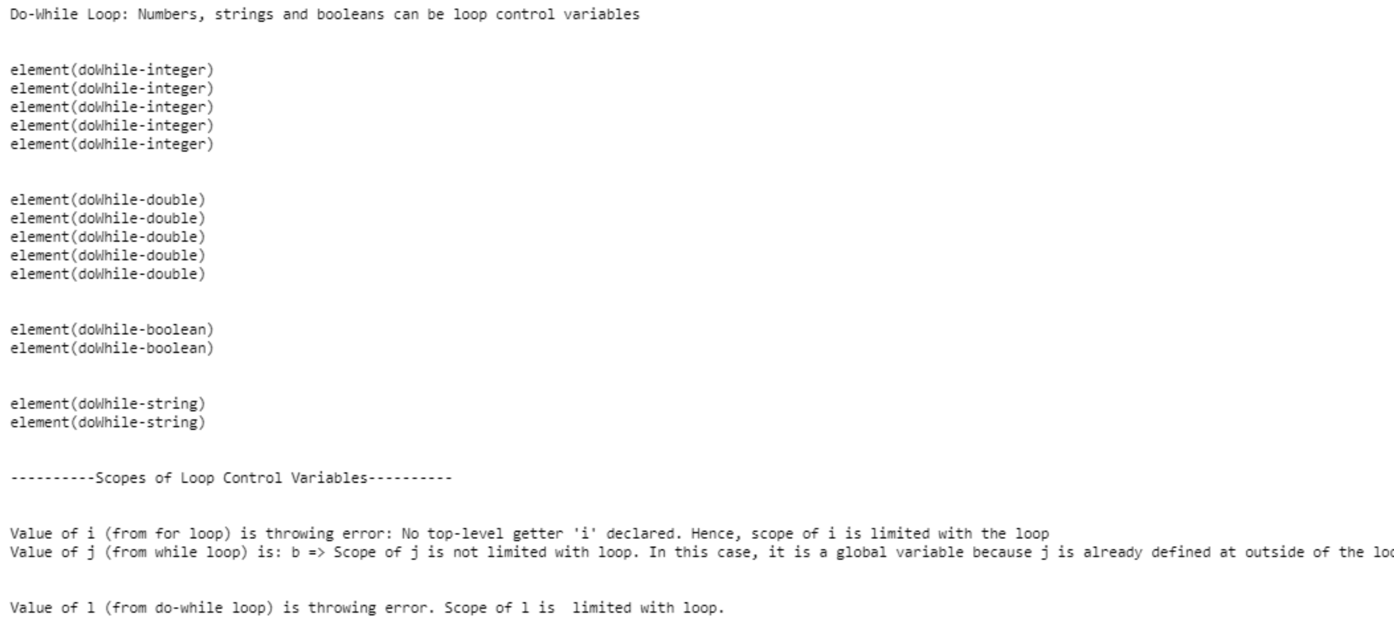
```
//Control for scope of value j
while (j != "b")
{
    j="b";
}
print("Value of j (from while loop) is: " + j + " => Scope of j is not limited with
    loop. In this case, it is a global variable because j is already defined at
    outside of the loop.");
print("\n");

//Control for scope of value l
//do
//{
    //print("element(doWhile-boolean)\n");
    //print("element(doWhile-boolean)\n");
//    var l =false;
//} while (l != false);

print("Value of l (from do-while loop) is throwing error. Scope of l is  limited with
    loop.");
print("\n");

print("----------Changing Loop Control Variables & Parameters----------\n\n");

//Prints just 1 time instead of 5 which shows that loop control variable can be
    changed in the loop and change can affect loop control.
for(int i = 0; i < 5; i++)
{
    print("element(variable change control)");
    i=4;
}
```



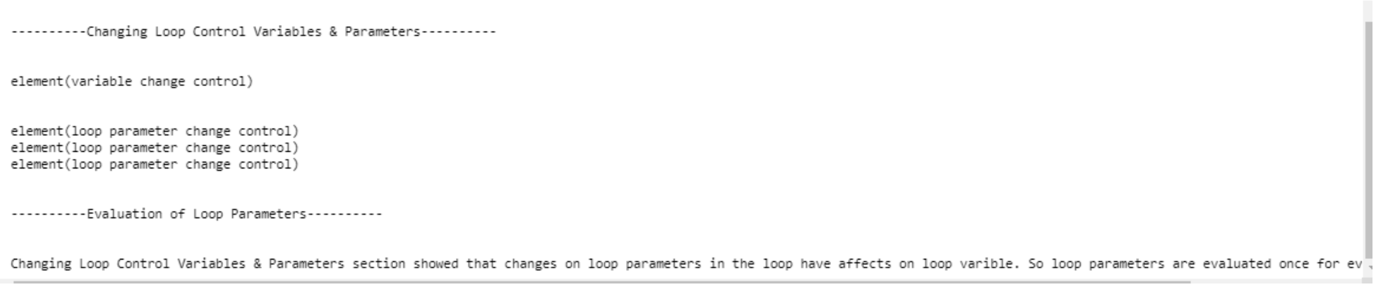
```
//prints 3 times instead of 5 which shows that loop parameters can be changed in the
    loop and change can affect loop control.
int m=5;
for(int i = 0; i < m; i++)
{
    print("element(loop parameter change control)");
    m = 3;
}

print("\n");

print("----------Evaluation of Loop Parameters----------\n\n");

print("Changing Loop Control Variables & Parameters section showed that changes on
    loop parameters in the loop have affects on loop varible. So loop parameters are
    evaluated once for every iteration in Dart. ");
```



```
----------Types of Loop Control Variables----------

For Loop: Integers and doubles can be loop control variable

element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)
element(for - integer)


element(for - double
element(for - double
element(for - double
element(for - double
element(for - double


For-in Loop: All data types can be loop control variable

foundLetter
foundLetter
foundLetter
foundLetter
foundLetter


While Loop: Integers, doubles, strings and booleans can be loop control variables


element(while-integer)
element(while-integer)
element(while-integer)
element(while-integer)
element(while-integer)





element(while-double)
element(while-double)
element(while-double)
element(while-double)
element(while-double)


element(while-boolean)
element(while-boolean)


element(while-string)
element(while-string)
```



```
Do-While Loop: Numbers, strings and booleans can be loop control variables


element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)
element(doWhile-integer)


element(doWhile-double)
element(doWhile-double)
element(doWhile-double)
element(doWhile-double)
element(doWhile-double)


element(doWhile-boolean)
element(doWhile-boolean)


element(doWhile-string)
element(doWhile-string)

----------Scopes of Loop Control Variables----------


Value of i (from for loop) is throwing error: No top-level getter 'i' declared. Hence, scope of i is limited with the loop
Value of j (from while loop) is: b => Scope of j is not limited with loop. In this case, it is a global variable because j is already defined at outside of the loo


Value of l (from do-while loop) is throwing error. Scope of l is  limited with loop.
```



```
----------Changing Loop Control Variables & Parameters----------


element(variable change control)


element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)


----------Evaluation of Loop Parameters----------


Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop have affects on loop varible. So loop parameters are evaluated once for ev
```

5. Rust

*Details are given in example program.

- **What are the types of loop control variables?**

  - **For:** Integer
  - **While:** Integer, Float, String and Boolean

- **What are the scopes of loop control variables?**

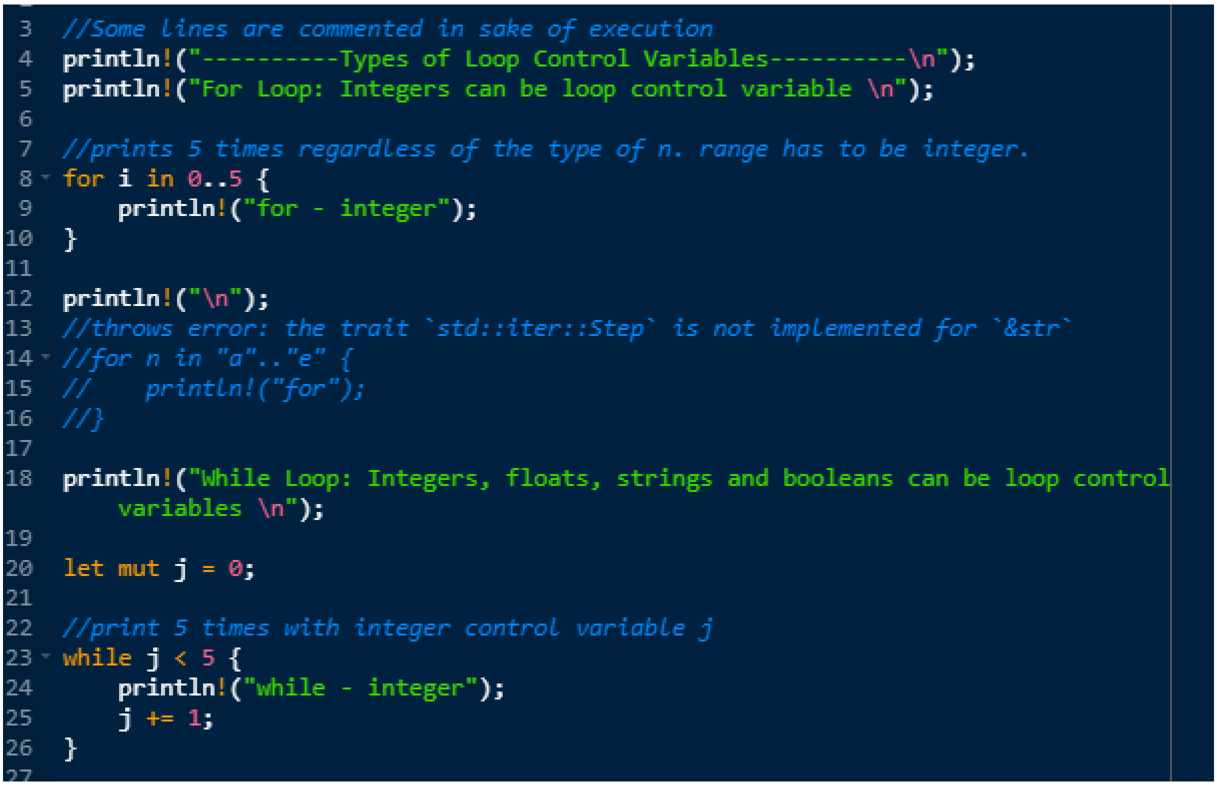
Scope of loop control variables of for loop is limited with the loop and it is not visible in global scope.

- **Is it legal for the loop control variable or loop parameters to be changed in the loop, and if so, does the change affect loop control?**

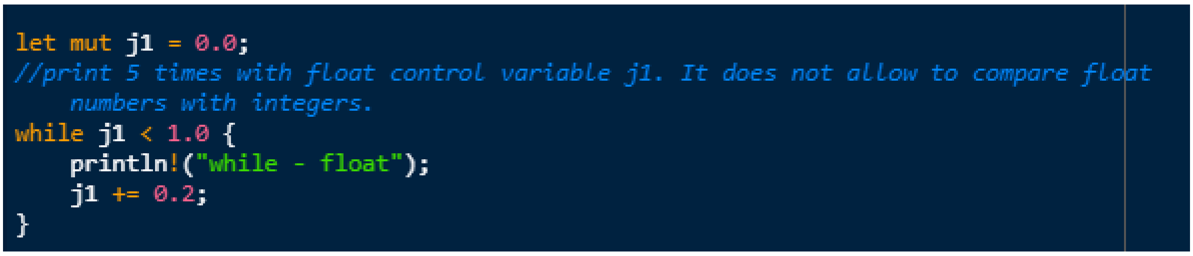
  No, it is not legal for the loop control variable or loop parameters to be changed in the loop.

- **Are the loop parameters evaluated only once, or once for every iteration?**

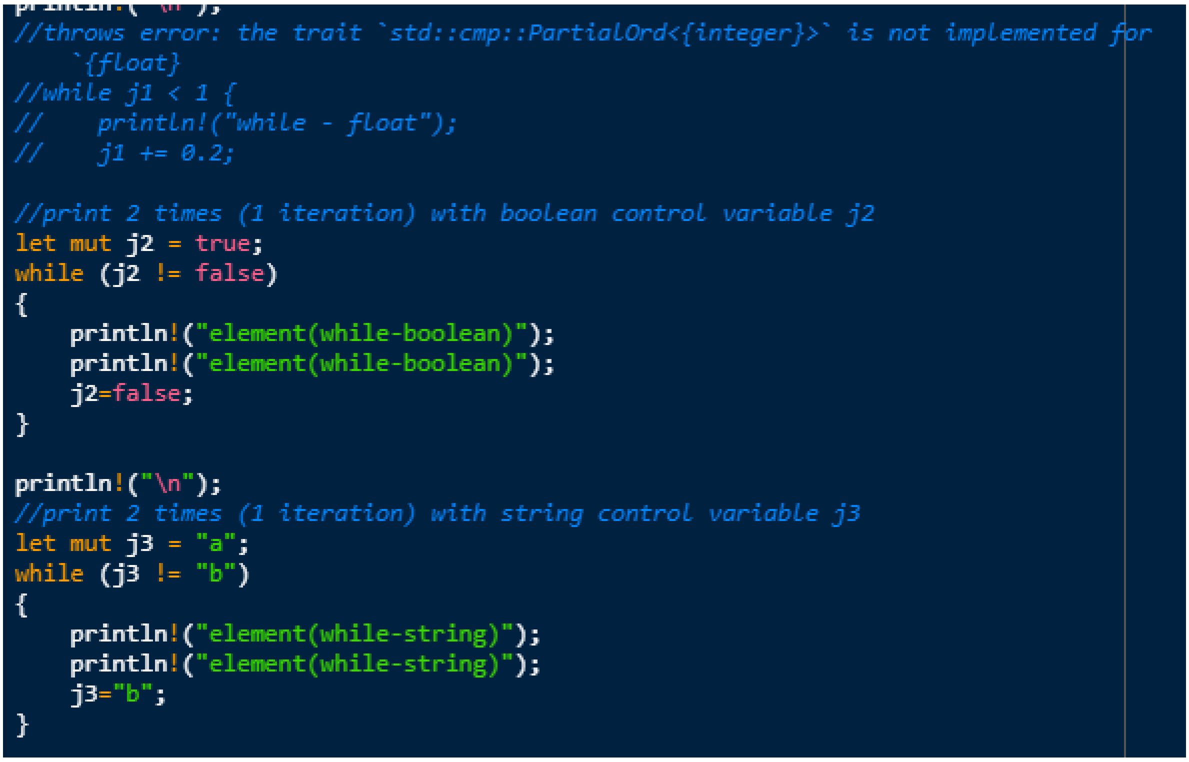
  Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop do not have any affect on loop. So loop parameters are evaluated only once before the loop in Rust.



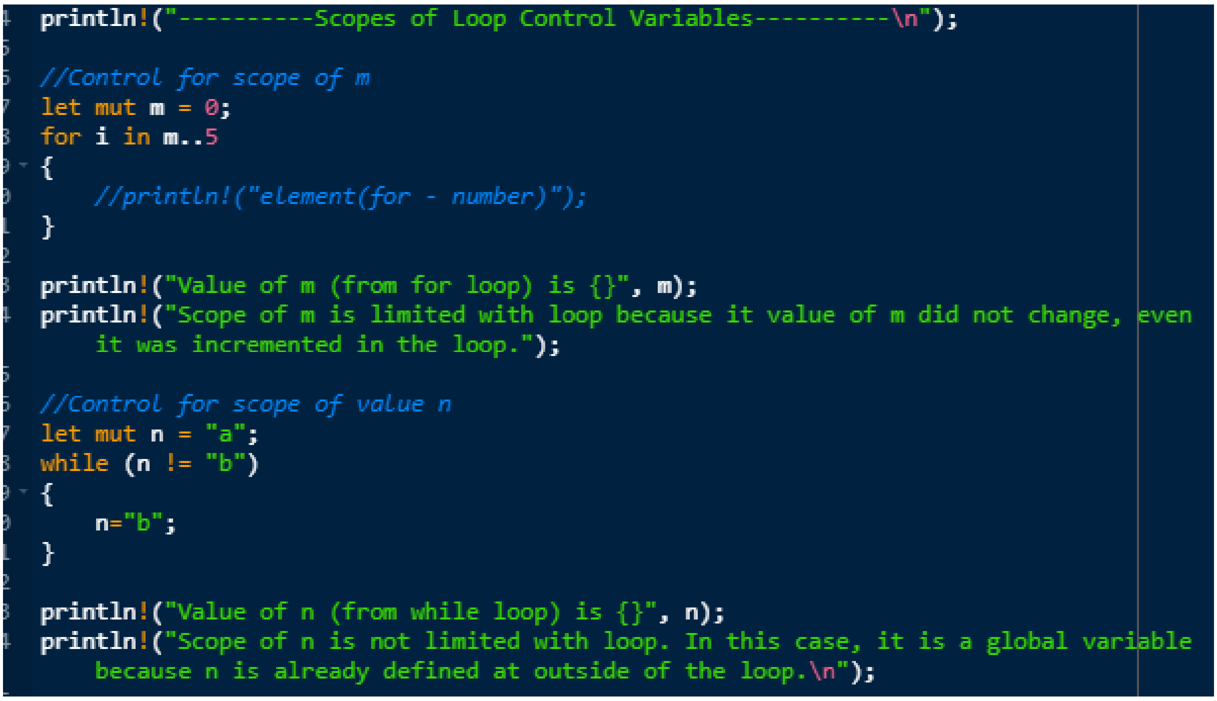
```rust
//Some lines are commented in sake of execution
println!("----------Types of Loop Control Variables----------\n");
println!("For Loop: Integers can be loop control variable \n");

//prints 5 times regardless of the type of n. range has to be integer.
for i in 0..5 {
    println!("for - integer");
}

println!("\n");
//throws error: the trait `std::iter::Step` is not implemented for `&str`
//for n in "a".."e" {
//    println!("for");
//}

println!("While Loop: Integers, floats, strings and booleans can be loop control variables \n");

let mut j = 0;

//print 5 times with integer control variable j
while j < 5 {
    println!("while - integer");
    j += 1;
}
```



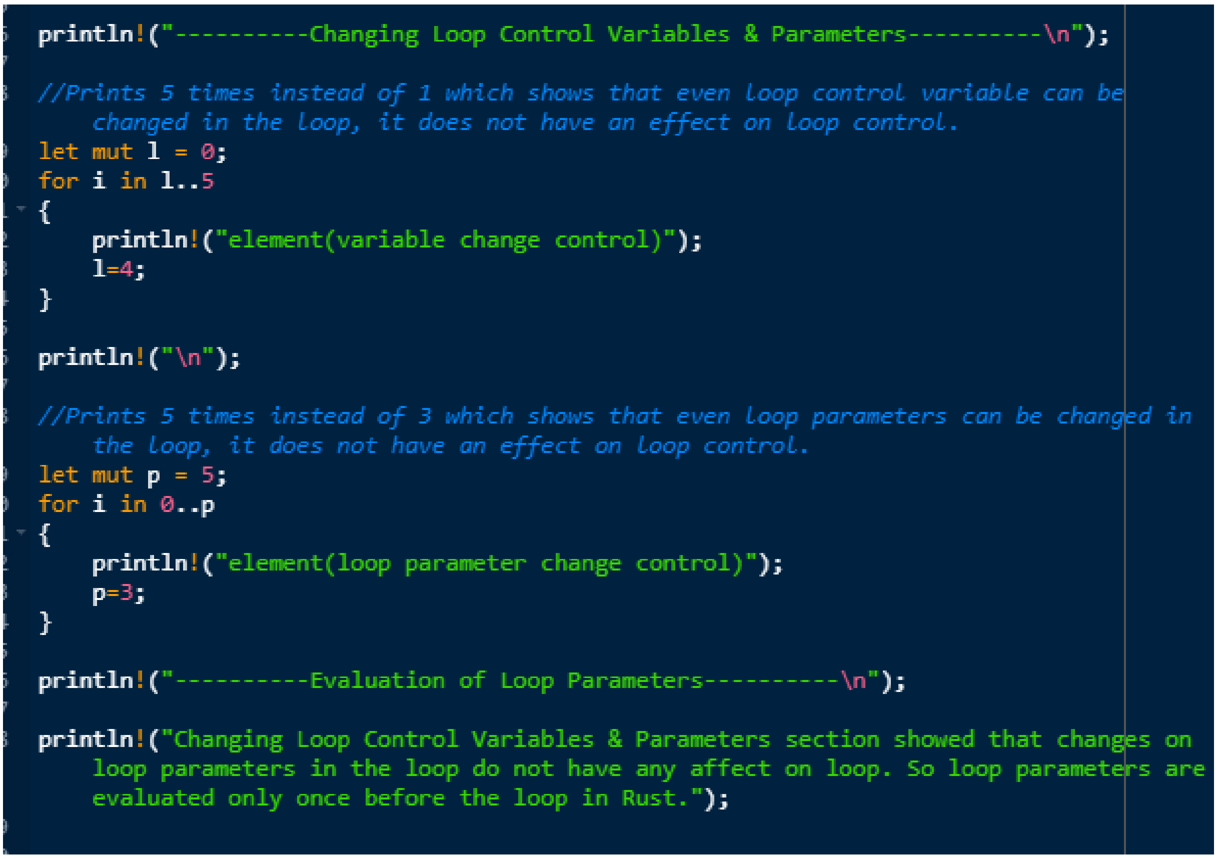
```rust
let mut j1 = 0.0;
//print 5 times with float control variable j1. It does not allow to compare float numbers with integers.
while j1 < 1.0 {
    println!("while - float");
    j1 += 0.2;
}
```



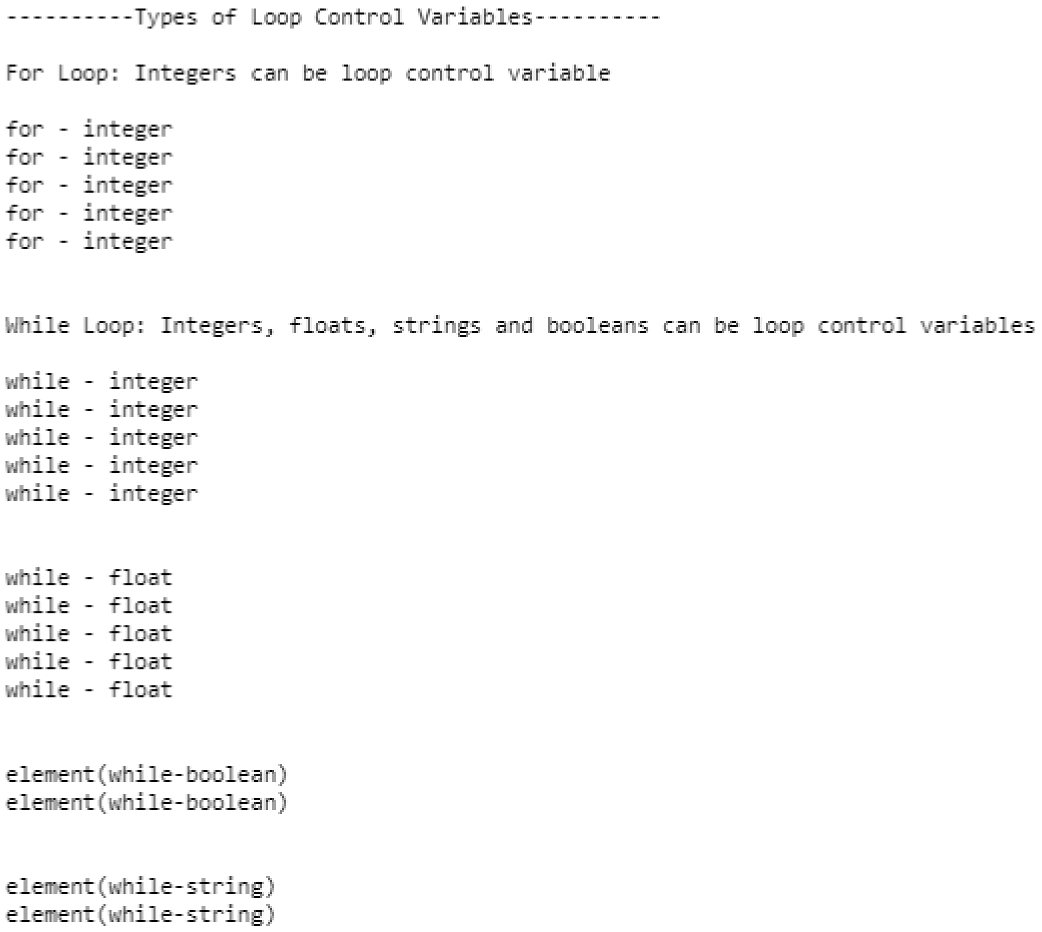
```
println!("\n");
//throws error: the trait `std::cmp::PartialOrd<{integer}>` is not implemented for
    `{float}`
//while j1 < 1 {
//    println!("while - float");
//    j1 += 0.2;

//print 2 times (1 iteration) with boolean control variable j2
let mut j2 = true;
while (j2 != false)
{
    println!("element(while-boolean)");
    println!("element(while-boolean)");
    j2=false;
}

println!("\n");
//print 2 times (1 iteration) with string control variable j3
let mut j3 = "a";
while (j3 != "b")
{
    println!("element(while-string)");
    println!("element(while-string)");
    j3="b";
}
```



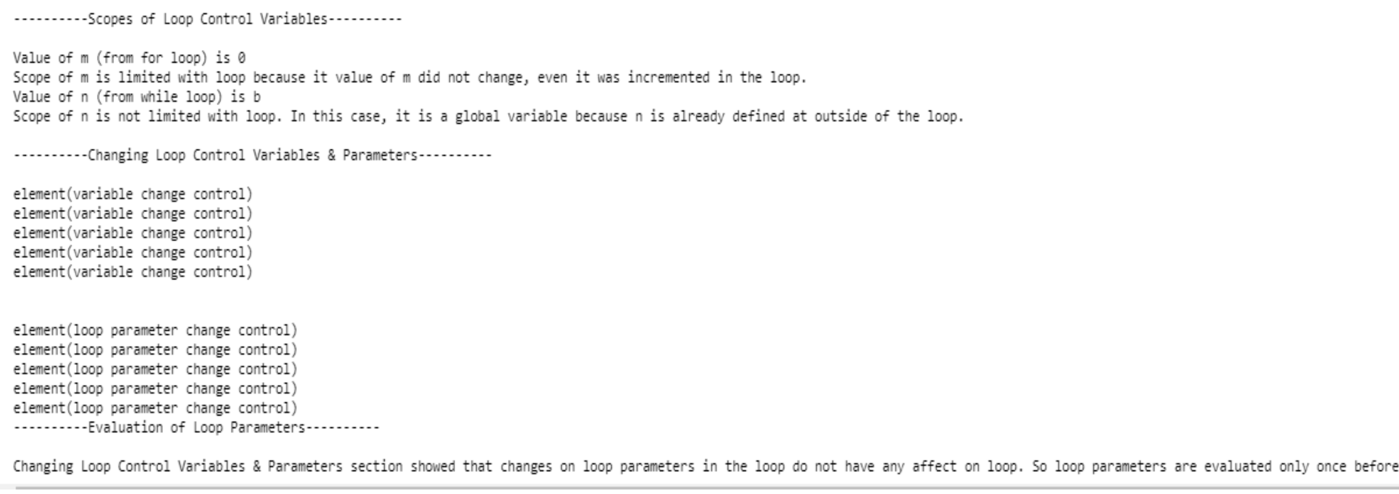
```
println!("----------Scopes of Loop Control Variables----------\n");

//Control for scope of m
let mut m = 0;
for i in m..5
{
    //println!("element(for - number)");
}

println!("Value of m (from for loop) is {}", m);
println!("Scope of m is limited with loop because it value of m did not change, even
    it was incremented in the loop.");

//Control for scope of value n
let mut n = "a";
while (n != "b")
{
    n="b";
}

println!("Value of n (from while loop) is {}", n);
println!("Scope of n is not limited with loop. In this case, it is a global variable
    because n is already defined at outside of the loop.\n");
```



```
println!("----------Changing Loop Control Variables & Parameters----------\n");

//Prints 5 times instead of 1 which shows that even loop control variable can be
    changed in the loop, it does not have an effect on loop control.
let mut l = 0;
for i in l..5
{
    println!("element(variable change control)");
    l=4;
}

println!("\n");

//Prints 5 times instead of 3 which shows that even loop parameters can be changed in
    the loop, it does not have an effect on loop control.
let mut p = 5;
for i in 0..p
{
    println!("element(loop parameter change control)");
    p=3;
}

println!("----------Evaluation of Loop Parameters----------\n");

println!("Changing Loop Control Variables & Parameters section showed that changes on
    loop parameters in the loop do not have any affect on loop. So loop parameters are
    evaluated only once before the loop in Rust.");
```



```
----------Types of Loop Control Variables----------

For Loop: Integers can be loop control variable

for - integer
for - integer
for - integer
for - integer
for - integer


While Loop: Integers, floats, strings and booleans can be loop control variables

while - integer
while - integer
while - integer
while - integer
while - integer


while - float
while - float
while - float
while - float
while - float


element(while-boolean)
element(while-boolean)


element(while-string)
element(while-string)
```



```
----------Scopes of Loop Control Variables----------

Value of m (from for loop) is 0
Scope of m is limited with loop because it value of m did not change, even it was incremented in the loop.
Value of n (from while loop) is b
Scope of n is not limited with loop. In this case, it is a global variable because n is already defined at outside of the loop.

----------Changing Loop Control Variables & Parameters----------

element(variable change control)
element(variable change control)
element(variable change control)
element(variable change control)
element(variable change control)


element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)
element(loop parameter change control)
----------Evaluation of Loop Parameters----------

Changing Loop Control Variables & Parameters section showed that changes on loop parameters in the loop do not have any affect on loop. So loop parameters are evaluated only once before
```

6. Best Language for Counter-Controlled Loops

Counter-controlled loops are the most used structure in application development that we see in every program. Even in different structures or names, there are counter-controlled loops in all modern programming languages. While doing homework, I had the opportunity to research and analyze the counter-controlled loop structures of five different languages. All of them have both advantages and disadvantages for loop handling. If I consider my own programming habits, my expectations from a loop structure is that it should be have good writability and I should perform my loop operations in the simplest way. From my perspective, I think Python is the best for loop operations. Because of its loop syntax, it is very easy to create loops and manage loop parameters of it. Also it has various functions which avoid thinking of complex loop algorithms and there is usually no need to write nested loops. Beside of the default functions, also it provides thousands of available functions via libraries. It can have reliability and performance issues but these are programming related issues and should not be evaluated in the scope of counter-controlled loops. So I can say that the best language is Python when I only consider the issue of loops.

7. Learning Strategies

The homework is mostly consisting of experimental sections. Before that, it was necessary to learn the syntax of languages. So, different learning practice strategies are required for these two different concepts.

First of all, the all sources that I used are on the internet. Internet offers the fastest and most efficient learning opportunity in programming-related subjects. But even though all the sources are on the internet, I used different approaches for the experimental sections and understanding the syntax.

Even there are similarities; five different languages have their own unique syntax. There are some websites called *tutorialspoint.com*, *w3schools.com* and *geekforgeeks.com* which clearly denotes the syntax of JavaScript, Python, PHP and Dart. All of these websites are stated in references. I used these websites for faster understanding of the syntax of these four languages. However there is not a third party source which presents deep information about Rust. There is some information in *tutorailspoint.com* about Rust but it is not sufficient for the homework. So I used official documentation of Rust for learning its syntax which is published on official website. In addition to syntax, I also used the sources above to figure out what type of loops exist in the language and what are the primitive data types of the language.

The experimental part of the homework consists of questions about, type of loop-control variables, scope check, change of loop parameters and evaluation of loop parameters. For understanding the type of loop control variables, I tried all primitive data types in the loops to see if they are valid. For checking the scope of loop parameters, I tried to print value of the variables at outside of the loop. I change the values of loop parameters in the loop to understand their effect on counting and loop termination. Results of change of loop parameters experiments were already giving the answer for evaluation of loop parameters.

8. References

[1] Dart Programming - Loops. (n.d.). Retrieved from https://www.tutorialspoint.com/dart_programming/dart_programming_loops.htm

[2] JavaScript For Loop. (n.d.). Retrieved from https://www.w3schools.com/js/js_loop_for.asp

[3] Python For Loops. Retrieved from https://www.geeksforgeeks.org/python-for-loops/

[4] PHP - Loop Types. (n.d.). Retrieved from https://www.tutorialspoint.com/php/php_loop_types.htm

[5] The Rust Programming Language. (n.d.). Retrieved from https://doc.rust-lang.org/1.29.0/book/first-edition/loops.html