CS315-HW2

Name: Elif Gülşah Kaşdoğan

ID: 21601183

Section: 3

For loops are referred as counter controlled loops in general. Termination of a for loop depends on check condition and update made in each iteration. Some languages use for/in or foreach loops for iterating through lists or structures compatible to lists. For and for/in loops can be converted to while loops generally and vice versa. I am not sure if we can say a for/in and for each loops are counter controlled loops but I mentioned about these in some of the programming languages since there are tradeoffs between these loops and for loops at some points.

Example codes are written in each programming language. For types of loop control variables, one for loop is written for each type to check their validity. For scoping, variable's visibility after loop termination is checked for global scope and function scope. Loop parameters are changed inside the loop wherever possible and results observed to see how the loop is affected. And lastly, I wrote functions which returns integers to use these integers for termination condition and I printed a statement inside the function. If statement is printed more than once this means for loop makes evaluation once for each iteration, if it is printed once this means evaluation is done only once.

To form a more readable report rather than including all code, I included important parts of code and explained.

Python:

Python has for and for/in loops.

What are the types of loop control variables?

1. Integers, characters, enumeration are allowed in counter controlled loops in Python. Floating point numbers cannot be used directly however list structure can be created in a manner similar to for loop:

```
floating = [x/10.0 for x in range (29,35,1)]
print(floating)
```

floating is a list of numbers and print statement displays these numbers by iterating the list one by one.

What are the scopes of loop control variables?

2. If loop is not in a function but global, scope of the loop variable is not limited with the loop, loop variable can be used after loop terminates. It will have the value which is lastly iterated in loop. Variable's scope will be the scope where for loop exists. However if the loop is inside a function loop variable's scope will be that function.

```
string = "gulsah"
result = " "
for char in string:
    if char == 'l':
        result += char.upper()
    else:
        result += char

print (result) #guLsah --> l will be uppercase after termination
print(char) #h --> lastly iterated element
```

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?

3. Python does not refer to loop variable inside the loop when you use its name directly. It shadows the loop variable thus it is not possible to change loop variable inside the loop. While loop can be used to perform an operation like this if necessary rather than for loop.

```
for x in range(3):
    x = x*10
    print(x)
```

Output: 0, 10, 20

If I could have changed the loop variable loop would terminate after printing 0.

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

4. In for in range functions in Python, loop in form: for x in range(item1, item2) item2 will be evaluated once, item1 will be used for initialization and x will be incremented by one for each iteration as long as loop continues.

If we have a for statement like the following:

for k in range(0, func(0)):

func(0) will be evaluated once in for loop, k will be incremented by one in each iteration and will be compared to func(0)'s result.

Code:

```
def numberGenerator(num):
    print("I am evaluated")
    return num + 5

#for k in range(0, numberGenerator(k)): --> k will not be accepted as parameter

for k in range(0, numberGenerator(0)):
    print(k)
```

Output:

I am evaluated

0

1

2

3

4

If we had multiple evaluations I am evaluated would be printed after each iteration of for loop.

Javascript:

JavaScript has for and for/in loops.

Counter controlled loop is in form:

for ([initialExpression]; [condition]; [updateExpression])

statement

all loop parameters are optional in JavaScript, incrementation can be done inside loop, condition check can be done inside the loop. But general convention is to use loop as in given structure.

What are the types of loop control variables?

1. Integers and floating points are allowed in JavaScript. However floating point in for loop variables may give wrong results and should be avoided. Characters are not used directly, each character has a character code similar to ASCII representation and can be retrieved by using char.charCodeAt(0) statement, for loop iteration can be done by using this kind of conversion. Or a character stream can be iterated by for/in loop.
    a. Enumarations can be used in for/in loop

```
for (let item in EnumClassName) {
  console.log( item );
}
```

    b.

```
var i;
for (i = 0; i < 2.1; i= i+0.1) {
```

```
        result1 += i.toString(10) + " ";
    }
```

Code b is valid but problematic, it is commented in executable code.

What are the scopes of loop control variables?

2. Javascript does not have block scope concept, it uses function scope. Loop variable uses its function's scope, variable is global if it is not encapsulated with a function. Last value of the for loop which makes termination condition false and causes loop to end will be stored in variable. This value is visible from its scope.

```
var loopVar1 = 9;
function func(){
        for(var loopVar1 = 0; loopVar1 < 5; loopVar1++){
        }
        //loopVar1 == 5
}
//loopVar1 == 9 because scope of loop variable is func.
```

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?

3. Loop variable can be changed inside the for loop, but not in for/in loop. For/in loop iterates thorough the given list or list like structure.

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

4. Initialization will be done once, condition evaluation will be done once for every iteration and updateExpression will be executed once as long as loop continues in for loops. for(var i = 0; i < array.length; ++i) → array.length will be evaluated once for each iteration or similarly if we have a statement like i < a+b operation a+b will be performed in each iteration.

```
function randomGen(){
        result4 += "<br> I am evaluated ! <br>";
        return 15;
}
```

```
for(var i = 10; i<randomGen(); i++){
        result4 += i + " ";
}
result += "<br>" + result4;
```

We will see “I am evaluated !“ statement 5 times as output on browser.

Perl:

for and foreach can be used as synonyms in Perl.

for is in form:

```
for (INITIALIZE; TEST; STEP) {
  BODY;
}
```

foreach is in form:

```
foreach my $i (0..9) {
  print "$i\n";
}
```

What are the types of loop control variables?

1. Integers, characters and floats are allowed in Perl for loop. Enumarations are not used in for loop because it may cause problems and unnecessary complexity. It is technically possible but gives unwanted results. Built-in function “use” is used by programmers in general.

```
#integer loop variable
for($i= 0; $i < 5; $i++){
   print "integer = $i\n";
}

#float as loop variable
for($i = 1.0 ; $i < 5.0; $i = $i + 0.8)
{
   printf "float =%.2f\n", $i;
}
#chars as loop variable
for(my $char = 'a'; $char le 'k'; $char++){
```

```
    print "char = $char\n";
}
```

What are the scopes of loop control variables?

2. If we declare a loop variable with my keyword, it will use lexical scoping. This will make variable invisible from outside the lexical scope. However if there are no lexical declarations in scope, loop variable will be global, it uses dynamic scoping.

```
my $var1 = -1; #lexical scoping
$var2 = -2;

for($var1= 0; $var1 < 3; $var1++){
   print "integer = $var1\n";
}
print "scope1 = $var1\n"; #uses dynamic scoping,visible

for(my $char2 = 'a'; $char2 le 'c'; $char2++){
   print "char = $char2\n";
}
print "scope3 = $char2\n"; #lexical scoping, invisible

for($var2 = 0; $var2 < 3; $var2++){
   print "integer = $var2\n";
}
print "scope1 = $var2\n"; #uses dynamic scoping,visible

sub scopeCheck(){
   for($var2 = 10; $var2 < 12; $var2++){
      print "var = $var2\n";
   }
}
scopeCheck();
print "var2 is now $var2\n"; #output: 12 not -2
#perl does not do shadowing for globals.
```

Output:

integer = 0

integer = 1

integer = 2

scope1 = 3

char = a

char = b

char = c

scope3 =

integer = 0

integer = 1

integer = 2

scope1 = 3

var = 10

var = 11

var2 is now 12

As a convention Perl programmers generally declare loop variables with keyword my if otherwise is not specifically needed.

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?

3. Loop variables can be changed within the loop. Loop control will be done as usual with the current value of the loop control variable.

```
#can we change loop variables within loop
for($i = 0; $i < 10; $i++){
  $i++;
  print "loop variable = $i\n";
```

Output:

loop variable = 1

loop variable = 3

loop variable = 5

loop variable = 7

loop variable = 9

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

4. Loop parameters will be evaluated once for each iteration. Perl has a control statement "redo" that restarts code block however conditional will not be evaluated again if redo statement is used.

for($i = 0; $i < evalFunc(); $i++) → evalFunc is evaluated each time.

Code:

```
#Evaluation of loop variables
sub evalFunc(){
   print("I am evaluated!\n");
   return 5;
}

for($i = 0; $i < evalFunc(); $i++){

}
```

Output:

I am evaluated!

I am evaluated!

I am evaluated!

I am evaluated!

I am evaluated!

I am evaluated!

PHP:

PHP for lops are in form:

```
for (init counter; test counter; update counter) {
   code to be executed;
}
```

In first statement we initialize loop variable, test counter becomes false or true after each iteration and loop terminates if it is false.

PHP has also for each built in function to iterate through a list of elements.

What are the types of loop control variables?

1. Loop variables in PHP can be integers, floats, characters. Loop structure will be limited when working with characters, “not equals !=” should be used instead < or > and loop variable can only be incremented by one.

   Strings can be used in for loop as well, I have found an example where we can count dates by converting them to strings. But since it is a direct conversion from string to date and date to string there is no range check of dates, it is programmer’s responsibility to enter valid date.

What are the scopes of loop control variables?

2. Most of the variables in PHP have a single scope, meaning they can be reached from anywhere after declaration and can also be used. Loop variables are still available after termination if they are not encapsulated with a function.

   When a loop is constructed inside a function, if we use a loop variable having same name with the global variable there will be shadowing and global variable will not be affected. We will see that the loop variable will retain its last value if the loop is not inside a function.

   Example Code:

   ```
   $var = 8;
   ```

```
$array = array(1,4,5,6,99);
function iter($list)
{
  foreach($list as $var){
     echo " $var";
  }
}
iter($array);
echo "<br> $var <br>"; //8 --> will retain its value because this var is not the same as we used
in function

foreach($array as $var){
     echo " $var";
}
echo "<br> $var <br>"; //99
```

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?

3. Yes, it is legal to change loop variable inside the loop, loop control will be evaluated according to this change. It is also possible to change loop variable of foreach loop but since it acts like an iterator changing loop variable will not affect loop. More precisely, if you increment variable of foreach loop it will not skip elements, number of printed elements will be same as before, their values will be changed.

```
foreach($array as $var){
  //var: 1 4 5 6 99
  $var++;
  echo " $var";
//It  would be 1 5 99 if the loop is effected
// Output: 2 5 6 7 100
}
```

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

4. For loops test counter(second parameter) will be evaluated once for every iteration, termination condition also will be evaluated once for every iteration, initialization will be done only once.

   ```
   for ($i = 0; $i <= funct() $i++) {   //body  }
   ```

   in this loop funct() will be evaluated once for each iteration, avoiding this speeds up code. Something like the following can be done to avoid this:

   ```
   $check = funct();
   for ($i = 0; $i <= $check; $i++) { //body  }
   ```

   Code:

   ```
   function numGen(){

    echo "Hello\n";
    return 3;
   }

   for( $i = 0; $i<numGen(); $i++){
    echo $i;
   }
   ```

   Hello will be printed 3 times. It would be printed once like in Python example if it was evaluated once.

Conclusion:

I think languages which are doing loop parameter evaluation only once are more preferable because they will be faster, I would prefer Python over other three languages because of that reason. Not seeing termination condition of loop straightforward might make debugging harder in Python; in this case my second choice would be JavaScript among three languages because Perl and PHP are less readable than JavaScript, low readability will also make debugging harder. It can be seen as there is a Readability-Performance tradeoff between JavaScript and Python. Python offers best counter controlled loops in my opinion because of the performance issues.