Bartu Atabek - 21602229
Section 02

CS 315 Homework 2 Report

Counter-Controlled Loops in Python

Since Python's native count controlled loop does not have the traditional C-style for loop structure but instead it has structure similar to for each loop.

1. What are the types of loop control variables?

Considering the object as the loop control variable it could be any iterable object type such as strings, arrays, tuples and dictionaries. On the other hand the loop variable takes on the values specified in the given range one for each iteration therefore they have the same type of the object which could be any type.

```
# for loop iterating over an array
array = ["apple", 3.14, 'bartu', 10, 2+5j, -9, "Steve".isalnum()]
for element in array:
    print(element),

print("\n")

# for loop iterating over dictionaries key-value pairs
car = {
  "brand": "Tesla",
  "model": "Roadster",
  "year": 2020
}

for key, value in car.items():
  print(key, value),

print("\n")

# for loop iterating over range of numbers between [1,10)
for number in range(1,10):
             print(number),

print("\n")

# for loop iterating over a string
for char in "Hello World":
    print(char),
```

Output:

```
apple 3.14 bartu 10 (2+5j) -9 True
('brand', 'Tesla') ('model', 'Roadster') ('year', 2020)
1 2 3 4 5 6 7 8 9
H e l l o   W o r l d
```

The first loop iterated each element in the array and printed them accordingly. Second loop iterated key value pairs the dictionary and outputted them as pairs, third loop printed the numbers ranging from 1 to 9 using the range function and finally the last loop printed each character in the string “Hello Word”.

2. What are the scopes of loop control variables?

Loop control variables; the loop variable and the object in Python can be accessed both locally in the loop scope and can be accessed out of the scope in the global namespace.

```
for number in range(1,10):
             print(number),

print("\nThe number was: " + str(number))
number += 1
print("The number is: " + str(number))
```

Output:

```
1 2 3 4 5 6 7 8 9
The number was: 9
The number is: 10
```

The loop outputs and prints the numbers ranging from 1 to 9 and after the loop is ended it accesses the loop variable prints it then increments it and prints it again showing that it could be accessed.

3. 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?

In Python it is legal for the loop control variables and the loop parameters to be changed in the loop and the change affects the loop control i.e. appending elements to a collection will change the number of iterations. Also, changing the loop variable will effect the output results too.

```
# changing loop variable inside the loop results all outputs to become '99'
for i in range(5):
    i = 99
    print("i is now: " + str(i))

print("\n")

# changing the object collection inside the array will reflect the new elements
# to be iterated as well.
isAppended = 0
colors = ["red", "blue", "green"] # should iterate 3 times
for color in colors:
    if isAppended == 0:
        colors.append("magenta") # appended element is also iterated
        isAppended = 1
    print(color),
```

Output:

```
i is now: 99
i is now: 99
i is now: 99
i is now: 99
i is now: 99

red blue green magenta
```

In the first loop changing loop variable inside the loop results all outputs to become ’99' in the second loop changing the object collection inside the array will reflect the new elements to be iterated as well.

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

In Python, the for construct is a single evaluation at the start is done which means that the creation of the loop objects are done once such as in range however, the loop variable is assigned in each iteration to the next element until the end of the collection is reached.

```
for number in range(13): # generate numbers 0 to 12 once at the start
    # calculation is done in every iteration
    print(str(number) + '\t' + str(2**number))
```

Output:

```
0       1
1       2
2       4
3       8
4       16
5       32
6       64
7       128
8       256
9       512
10      1024
11      2048
12      4096
```

The loop generate numbers 0 to 12 once at the start and calculation is done in every iteration and the result is outputted.

Counter-Controlled Loops in Perl

1. What are the types of loop control variables?

For the C-style for loops, the loop control variable could have integer or floating point type i.e double. For the foreach style for loop, character collections, strings, map, and arrays are accepted.

```perl
# for loop with integer type control variable
for (my $index = 1; $index <= 6; $index++) {
  print("Iterated $index times.\n");
}

print "\n";

# for loop with floating type control variable
my $count = 1;
for (my $index = 0.5; $index <= 5; $index += 0.5) {
  print("Iterated $count times.\n");
  $count++;
}

print("\n");

# foreach style for loop iterating over range of numbers [1,6]
my @range = (1..6);
for (@range) {
  print("$_ "); # default variable
}
```

Output:

```
Iterated 1 times.
Iterated 2 times.
Iterated 3 times.
Iterated 4 times.
Iterated 5 times.
Iterated 6 times.

Iterated 1 times.
Iterated 2 times.
Iterated 3 times.
Iterated 4 times.
Iterated 5 times.
Iterated 6 times.
Iterated 7 times.
Iterated 8 times.
Iterated 9 times.
Iterated 10 times.
```

1 2 3 4 5 6

The first loop prints index numbers throughout the loop using integers. Second loop iterates using floating type as the control variables and prints the number of iterations. The third loop is foreach style for loop iterating over range of numbers [1,6].

2. What are the scopes of loop control variables?
If my keyword exist before the control variable it cannot be accessed outside the for loop scope however, without it, it can be accessed out of the scope. In the case of foreach loop the control variables can be accessed depending on the scope they were defined also, the default variable cannot be accessed outside the for loop.

```
for ($index = 1; $index <= 6; $index++) {
  print("Index is: $index\n");
}

print("\nIndex was: $index");
$index = "Hello World";
print("\nIndex is now: $index\n");
print("\n");

my @a = (1..10);
for $i (@a) {
 print("Index is: $i\n");
}

print("\nIndex is: $i\n"); # explicit iterator cannot accessed outside the scope
print("Numbers are: @a\n");
```

Output:

```
Index is: 1
Index is: 2
Index is: 3
Index is: 4
Index is: 5
Index is: 6

Index was: 7
Index is now: Hello World

Index is: 1
Index is: 2
Index is: 3
Index is: 4
Index is: 5
Index is: 6
Index is: 7
Index is: 8
Index is: 9
Index is: 10
```

Index is:
Numbers are: 1 2 3 4 5 6 7 8 9 10

The loop outputs and prints the numbers ranging from 1 to 10 and after the loop is ended it accesses the loop variable prints it then increments it and prints it again showing that it could be accessed.

3. 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?

In Perl it is legal for to change the loop control variables or loop parameters in the loop for the C-style for loop and it affects the loop control.In the case of foreach style loop it is also allowed to change the loop variables and if the iterable object is changed it also affects the loop control but changing the explicit iterator would not result in any effects.

```
for ($index = 1; $index <= 6; $index++) {
  $index = 10;
  print("Index is: $index\n");
}

print("\n");

my $isAppended = 0;
my @colors = ("red", "blue", "green");
for my $color (@colors) {
  if ($isAppended == 0) {
        push @colors, ("magenta", "orange"); # appended element is also iterated
        $isAppended = 1
  }
  print("$color"," ");
}
```

Output:

Index is: 10
red blue green magenta orange

The first loop outputs only once because the control variable is changed and since it it bigger than 6 it causes the loop to finish. The second loop appends two new colors and it prints them accordingly.

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

In Perl loop parameters are evaluated once for every iteration in C-style, for the foreach styled loops a single evaluation at the start is done which means that the creation of the loop objects are done once such as in range however, the loop variable is assigned in each iteration to the next element until the end of the collection is reached.

```
sub counter_statement {
  print "Statement evaluated\n";
  my $limit = 5;
  return $limit;
}

for ($index = 0; $index < counter_statement(); $index++) {
}
```

Output:

```
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
```

The following loop outputs the print statements six times indicating that the loop parameter is evaluated once every iteration.

Counter-Controlled Loops in JavaScript

1. What are the types of loop control variables?

For the C-style for loops, the loop control variable could have integer or floating point types. For the for/in statement loops, iterable type objects are accepted as it loops through the properties of an object.

```
// for loop with integer type control variable
   for (var i = 0; i < 5; i++) {
     console.log(i);
   }

   console.log("\n");

   // for loop with floating type control variable
   var count = 1;
   for (var index = 0.5; index <= 5; index += 0.5) {
     console.log(count);
     count++;
   }

   console.log("\n");

   // for/in style for loop iterating over range of numbers [1,6]
   var person = {fname:"Steve", lname:"Jobs", age:56};
   for (var attribute in person) {
     console.log(person[attribute]);
   }
```

Output:

```
0
1
2
3
4
```

```
1
2
3
4
5
6
7
8
9
10

Steve
Jobs
56
```

The first loop outputs the number of iterations and the loop control variable is integer, second loop does the same thing but the variable has float type. Last loop iterates over the object and outputs it’s attributes.

2. What are the scopes of loop control variables?

In JavaScript for the C-style for loops and the for/in loops loop control variables are accessible both inside the for loop block and also they could be accessed outside. However, in the case where the 'let' keyword is used for the control variable it is only accessible within the loop block and cannot be accessed outside the scope.

```
// index can be accessed outside the loop scope
    var index = 5;
    for (; index < 10; index++) {
      console.log("Index is: " + index);
    }

    console.log("\n");
    console.log("After the loop index is: " + i);
    console.log("\n");

    // index can be accessed outside the loop scope
    for (var a = 0; a < 3; a++) {
      console.log("A is: " + a);
    }

    console.log("\n");
    console.log("After the loop A is: " + a);
    console.log("\n");

    // let keyword constraints the scope of the control variable to the inside
    // of the for loop block
    for (let b = 0; b < 3; b++) {
      console.log("B is: " + b);
    }

    // can't access b because it's out of scope
    console.log("\n");
    console.log("After the loop b can't be accessed");
    // console.log("\nB is: " + b);
```

<u>Output:</u>

```
Index is: 5
Index is: 6
Index is: 7
Index is: 8
Index is: 9

After the loop index is: 5

A is: 0
A is: 1
A is: 2

After the loop A is: 3

B is: 0
B is: 1
B is: 2

After the loop B can’t be accessed.
```

3. 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?

In JavaScript it is legal for to change the loop control variables or loop parameters in the loop for the C-style for loop and it affects the loop control. In the case of for/in style loop it is also allowed to change the loop variables but it would would not result in any effects

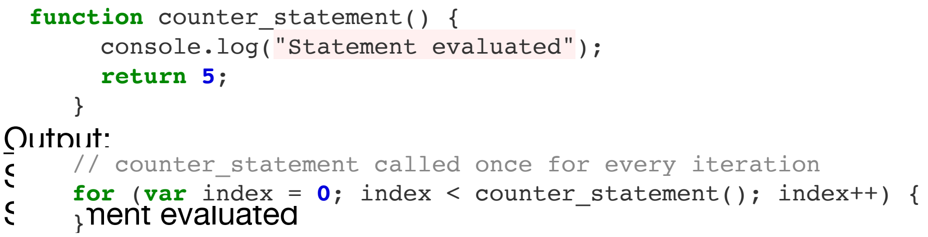
```
// changing control variable affects loop control
    var index = 1;
    for (; index < 6; index++) {
      index = 10;
      console.log("Index is: " + index + "\n");
    }

    console.log("\n");

    // appended element does not affect the control
    var isAppended = 0;
    var colors = ["red", "blue", "green"];
    for (var color in colors) {
      if (isAppended == 0) {
            colors.push("magenta", "orange");
            isAppended = 1
      }
      console.log(colors[color]);
    }
    console.log(colors); // new colors added but not displayed in the loop
```

Output:
Index is: 10

Red blue green

Red blue green magenta orange

4. Are the loop parameters evaluated only once, or once for every iteration?
In JavaScript loop parameters are evaluated once for every iteration in C-style, for the for/in styled loops a single evaluation at the start is done which means that the creation of the loop objects are done once however, the loop variable is assigned in each iteration to the next element until the end of the collection is reached.



```
function counter_statement() {
     console.log("Statement evaluated");
     return 5;
   }

   // counter_statement called once for every iteration
   for (var index = 0; index < counter_statement(); index++) {
   }
```

Output:

```
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
```

The following loop outputs the print statements six times indicating that the loop parameter is evaluated once every iteration.

Counter-Controlled Loops in PHP

1. What are the types of loop control variables?
For the C-style for loops, the loop control variable could have integer, floating point types or string. For the foreach type loops, array type is accepted as it loops through a block of code for each element in an array.

```
# for loop with integer type control variable
  for ($index = 1; $index <= 6; $index++) {
    echo "Iterated " . $index . " times.<br>";
  }

  echo "<br>";

  # for loop with floating type control variable
  $count = 1;
  for ($index = 0.5; $index <= 5; $index += 0.5) {
    echo "Iterated " . $count . " times.<br>";
    $count++;
  }

  echo "<br>";

  # for loop with string type control variables
  for ($index = "a"; $index <= "e"; $index++) {
    echo "Iterated " . $index . " times.<br>";
  }

  echo "<br>";

  # foreach style for loop iterating over array of numbers [1,5]
  $array = array(1, 2, 3, 4, 5);
  foreach ($array as $value) {
    echo "Value is $value <br>";
  }
```

Output:

```
Iterated 1 times.
Iterated 2 times.
Iterated 3 times.
Iterated 4 times.
Iterated 5 times.
Iterated 6 times.

Iterated 1 times.
Iterated 2 times.
Iterated 3 times.
Iterated 4 times.
Iterated 5 times.
Iterated 6 times.
Iterated 7 times.
Iterated 8 times.
Iterated 9 times.
Iterated 10 times.

Iterated a times.
Iterated b times.
Iterated c times.
Iterated d times.
Iterated e times.

Value is 1
Value is 2
Value is 3
Value is 4
Value is 5
```

2. What are the scopes of loop control variables?

In PHP for the C-style for loops and the for/in loops loop control variables are accessible both inside the for loop block and also they could be accessed outside.

```
 // index can be accessed outside the loop scope
  for ($index = 5; $index < 10; $index++) {
    echo "Index is: " . $index . " <br>";
  }

  echo "<br>";
  echo "After the loop index is: " . $index . " <br>";
  echo "<br>";

  // last element can be accessed outside the loop scope
  foreach ($array as $value) {
    echo "Value is $value <br>";
  }

  echo "<br>";
  echo "The last element is: " . $value . "<br>";
```

Output:

```
Index is: 5
Index is: 6
Index is: 7
Index is: 8
Index is: 9

After the loop index is: 10

Value is 1
Value is 2
Value is 3
Value is 4
Value is 5

The last element is: 5
```

3. 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?

In PHP it is legal for to change the loop control variables or loop parameters in the loop for the C-style for loop and it affects the loop control. In the case of foreach style loop it is also allowed to change the loop variables but it would would not result in any effects.

```
// changing control variable affects loop control
  for ($index = 1; $index < 6; $index++) {
    $index = 10;
    echo "Index is: " . $index . " <br>";
  }

  echo "<br>";

  // appended element does not affect the control
  // new colors added but not displayed in the loop
  $isAppended = 0;
  $colors = array("red", "blue", "green");
  foreach ($colors as $color) {
    if ($isAppended == 0) {
      array_push($colors, "magenta", "orange");
      $isAppended = 1;
    }
    echo "Color is: $color <br>";
  }

  echo "<br>";

  foreach ($colors as $color) {
    echo "$color "; // contains 'magenta' and 'orange'
  }
```

Output:

```
Index is: 10

Color is: red
Color is: blue
Color is: green

red blue green magenta orange
```

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

In PHP loop parameters are evaluated once for every iteration in C-style, for the for/in styled loops a single evaluation at the start is done which means that the creation of the loop objects are done once however, the loop variable is assigned in each iteration to the next element until the end of the collection is reached.

```
 function counter_statement() {
    echo "Statement evaluated<br>";
    $limit = 5;
    return $limit;
  }

  for ($index = 0; $index < counter_statement(); $index++) {
  }
```

Output:

```
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
Statement evaluated
```

Which language provides the best counter-controlled loop?

In my opinion java script offers the best counter controlled loop because of it’s versatility. It enables us write the for loop in many different ways which increases the it’s usability and it also offers c type for statements. The loop parameters can be constrained by the user if specified by the ‘let’ keyword. Also it has two control mechanism for the for each mechanism. Loop variables can be defined outside the loop. It’s structure increases readability and writability which are the language evaluation criterias.