LECTURE 6.1: CONTROL FLOW - LOOPS

BY LINA HAMMAD & AHMAD BARGHASH

The for loop in Kotlin iterates through anything that provides an iterator. In this lecture, you learn to create for loop, while and do...while (with the help of examples).

LOOPS IN KOTLIN

- Loop is used in programming to repeat a specific block of code until certain condition is met (test expression is false).

- Loops are what makes computers interesting machines. Imagine you need to print a sentence 50 times on your screen. Well, you can do it by using print statement 50 times (without using loops). How about you need to print a sentence one million times? You need to use loops.

- You will learn about three loops types:
  1. For
  2. While
  3. do..while

FOR LOOP

- Loop is such an invention that provides the flexibility to iterate or cycle though the elements of array, ranges, collections maps and so on (anything that provides an iterator). Like other programming languages, Kotlin also provides many kinds of Looping methodology, however, among them “For” is the most successful one.

- There is no traditional for loop in Kotlin unlike Java and other languages.

FOR LOOP SYNTAX

- If the body of the loop contains only one statement , it's not necessary to use curly braces { }:

```
for (item in collection) // ...
```

- The body can be a block.

```
for (item in collection) {
    // ...
}
```

SIMPLE FOR LOOP EXAMPLE

```
fun main() {
    for (i in 1..3) {
        println(i)
    }
}
```

The output is:

```
1
2
3
```

```
fun main() {
    for(n in 10..15){
        println("Loop: $n")
    }
}
```

The output is:

```
Loop: 10
Loop: 11
Loop: 12
Loop: 13
Loop: 14
Loop: 15
```

FOR LOOP EXAMPLE USING DOWNTO FUNCTION

- To iterate numbers in reverse order, use the **downTo** function instead of ...

```
fun main(){
    for (i in 4 downTo 1) print(i)
}
```

The output is:

```
4321
```

FOR LOOP EXAMPLE USING STEP FUNCTION

- It is also possible to iterate over numbers with an arbitrary step (not necessarily 1). This is done via the **step** function.

```
fun main() {
    for (i in 1..8 step 2) print(i)
    println("-----")
    for (i in 8 downTo 1 step 2) print(i)
}
```

The output is:

```
1357
-----
8642
```

FOR LOOP EXAMPLE USING UNTIL FUNCTION

- To iterate a number range which does not include its end element, use the **until** function:

```
fun main(){
    for (i in 1 until 10) { // i in [1, 10), 10 is excluded
        print(i)
    }
}
```

The output is:

```
123456789
```

FOR LOOP EXAMPLES

```
fun main() {
    print("for (i in 1..5) print(i) = ")
    for (i in 1..5) print(i)

    println()

    print("for (i in 5..1) print(i) = ")
    for (i in 5..1) print(i)

    println()

    print("for (i in 5 downTo 1) print(i) = ")
    for (i in 5 downTo 1) print(i)

    println()

    print("for (i in 1..4 step 2) print(i) = ")
    for (i in 1..5 step 2) print(i)

    println()

    print("for (i in 4 downTo 1 step 2) print(i) = ")
    for (i in 5 downTo 1 step 2) print(i)
}
```

The output is:

```
for (i in 1..5) print(i) = 12345
for (i in 5..1) print(i) =
for (i in 5 downTo 1) print(i) = 54321
for (i in 1..4 step 2) print(i) = 135
for (i in 4 downTo 1 step 2) print(i) = 531
```

ITERATING THROUGH A STRING

```
fun main() {
    var text= "Kotlin"
    for (letter in text) {
        println(letter)
    }
}
```

The output is:

```
K
o
t
l
i
n
```

```
fun main() {
    var text= "Kotlin"
    for (item in text.indices) {
        println(text[item])
    }
}
```

The output is:

```
K
o
t
l
i
n
```

IF INSIDE FOR EXAMPLE

```
fun main(){
    var myString = "Hello, this piece of code, is written using Kotlin."
    var numberOfChar:Int= 0
    var numberOfIChar:Int= 0

    for(i in 0..myString.length-1){

        println(myString[i])
        numberOfChar++


        if(myString[i] == 'i'){
            numberOfIChar++
        }
    }

    println("Number of character in the string = $numberOfChar")
    println("Number of \'i\' character in the string = $numberOfIChar")

}
```

IF INSIDE FOR EXAMPLE SOLUTION

The output is:

```
H
e
l
l
o
,

t
.
.
i
n
Number of character in the string = 46
Number of 'i' character in the string = 5
```

FOR LOOPS WITH USER INPUT

```
fun main(){
    println("please enter first number")
    var start = readLine()!!.toInt()

    println("please enter second number")
    var end = readLine()!!.toInt()

    for(i in start..end){
        println("i = $i")
    }
}
```

The output is:

```
please enter first number
12
please enter second number
20
i = 12
i = 13
i = 14
i = 15
i = 16
i = 17
i = 18
i = 19
i = 20
```

FOR LOOPS WITH USER INPUT

```
fun main(){
    println("please enter first number")
    var start = readLine()!!.toInt()
    println("please enter second number")
    var end = readLine()!!.toInt()

    for(i in start..end){
        if(i % 2 == 0)
        println("$i is even number")
    }

}
```

The output is:

```
please enter first number
1
please enter second number
10
2 is even number
4 is even number
6 is even number
8 is even number
10 is even number
```

KOTLIN WHILE LOOP

- While loop is used to iterate a block of code repeatedly as long as the given condition returns true. In this guide, we will learn how to use while loop with the help of examples.

- If you are familiar with while and do...while loops in Java, you are already familiar with these loops in Kotlin as well.

KOTLIN WHILE LOOP

- The syntax of while loop is:

```
while (testExpression) {
    // codes inside body of while loop
}
```

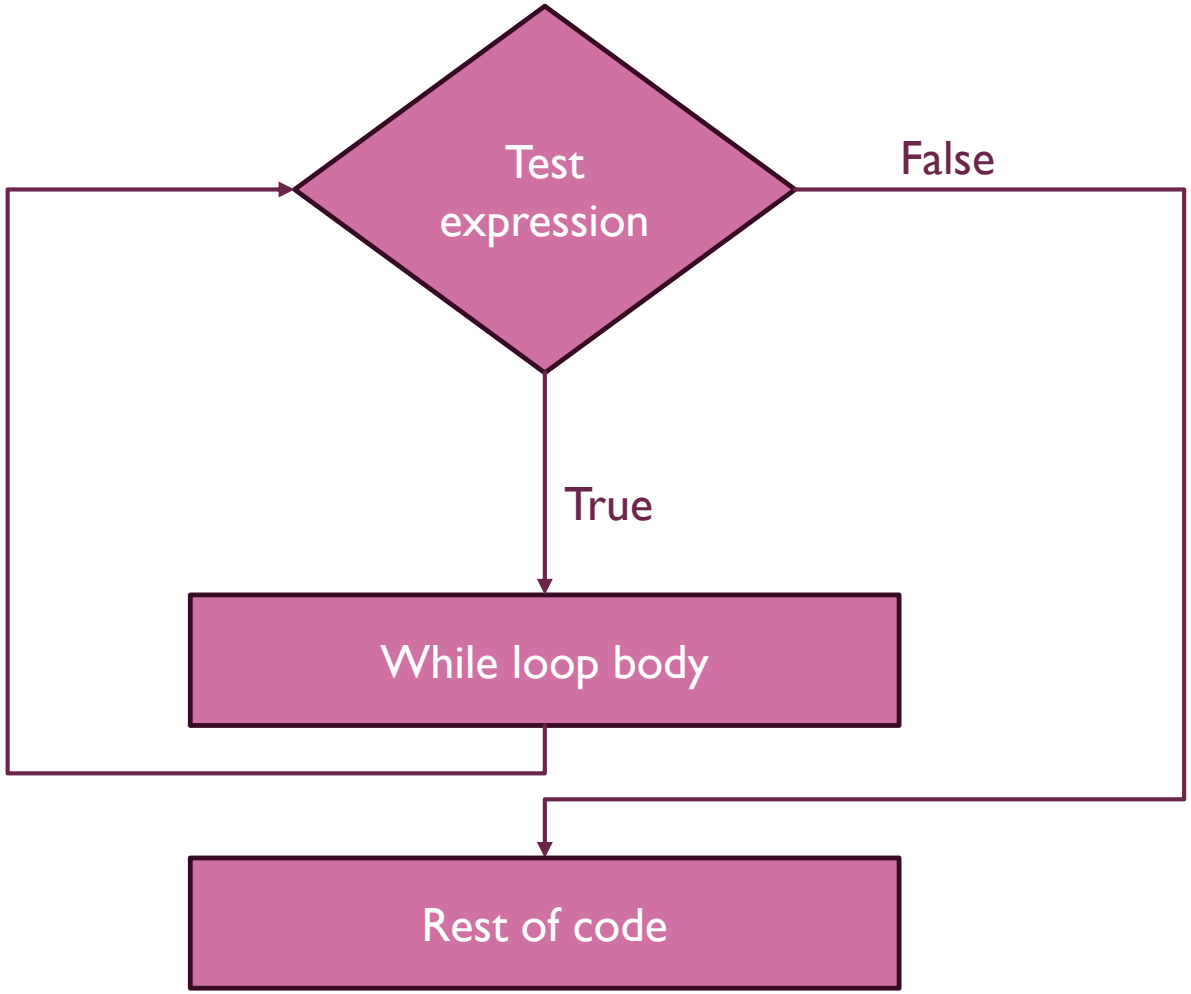
HOW WHILE LOOP WORKS?

- The test expression inside the parenthesis is a Boolean expression.

- If the test expression is evaluated to true:
  1. statements inside the while loop are executed.
  2. then, the test expression is evaluated again.
  3. This process goes on until the test expression is evaluated to false.

- If the test expression is evaluated to false:
  1. while loop is terminated.

FLOWCHART OF WHILE LOOP



EXAMPLE: KOTLIN WHILE LOOP

```
// Program to print line 5 times
fun main() {

    var i = 1

    while (i <= 5) {
        println("Line $i")
        ++i
    }

}
```

The output is:

```
Line 1
Line 2
Line 3
Line 4
Line 5
```

EXAMPLE: KOTLIN WHILE LOOP

- Notice, in the previous example, ++i statement inside the while loop. After 5 iterations, variable i will be incremented to 6. Then, the test expression i <= 5 is evaluated to false and the loop terminates.

- If the body of loop has only one statement, it's not necessary to use curly braces { }.

EXAMPLE: COMPUTE SUM OF NATURAL NUMBERS

```
fun main() {
    var sum = 0
    var i = 100
    while (i != 0) {
        sum += i     // sum = sum + i;
        --i
    }
    println("sum = $sum")
}
```

The output is:

```
sum = 5050
```

EXAMPLE: WHILE LOOP

```
fun main() {
    var x:Int = 0
    println("Example of While Loop--")

    while(x <= 10) {
        println(x)
        x++
    }
}
```

The output is:

```
Example of While Loop--
0
1
2
3
4
5
6
7
8
9
10
```

INFINITE WHILE LOOP

- If the condition specified in the while loop never returns false then the loop iterates infinitely and never stops such while loops are called infinite while loops. We should always avoid such situation while writing code. Lets see few examples of infinite while loop.

1. Since the condition is always true this will run infinitely.

```
while (true){
    println("loop")
}
```

INFINITE WHILE LOOP

2. In this while loop we are incrementing the counter num, the counter initial value is 10 and we are increasing it on every iteration, which means the specified condition num>=5 will always remain true and the loop will never stop.

```
var num = 10

while(num>=5){
    println("Loop: $num")
    num++
}
```

3. In this while loop we are incrementing the counter num, the counter initial value is 10 and we are increasing it on every iteration, which means the specified condition num>=5 will always remain true and the loop will never stop.

```
var num = 5

while(num<=10){
    println("Loop: $num")
    num--
}
```

KOTLIN DO...WHILE LOOP

- The do...while loop is similar to while loop with one key difference. The body of do...while loop is executed once before the test expression is checked.

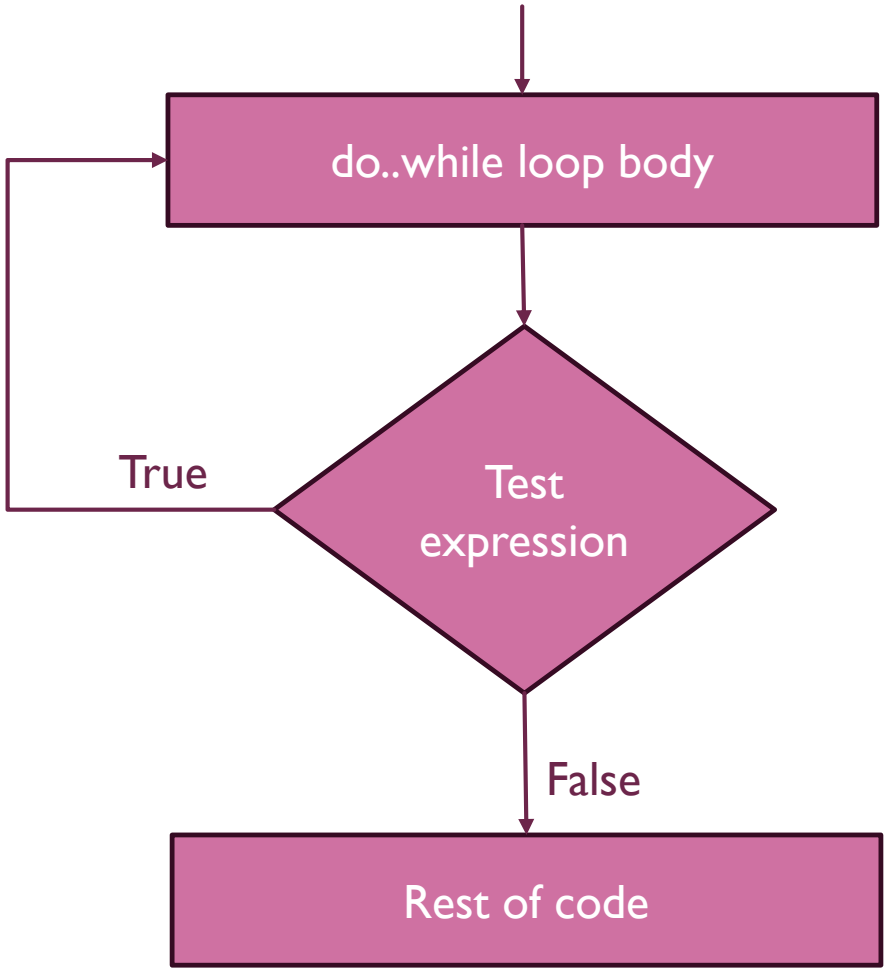
- Its syntax is:

```
do {
    // codes inside body of do while loop
} while (testExpression);
```

HOW DO...WHILE LOOP WORKS?

- The codes inside the body of do construct is executed once (without checking the testExpression). Then, the test expression is checked.
- If the test expression is evaluated to true, codes inside the body of the loop are executed, and test expression is evaluated again. This process goes on until the test expression is evaluated to false.
- When the test expression is evaluated to false, do..while loop terminates.

FLOWCHART OF DO...WHILE LOOP



EXAMPLE: KOTLIN DO...WHILE LOOP

- The program below calculates the sum of numbers entered by the user until user enters 0.
- To take input from the user, readline() function is used.

```
fun main( ) {
    var sum: Int = 0
    var input: String
    do {
        print("Enter an integer: ")
        input = readLine()!!
        sum += input.toInt()
    } while (input != "0")

    println("sum = $sum")
}
```

The output is:

```
Enter an integer: 4
Enter an integer: 3
Enter an integer: 2
Enter an integer: -6
Enter an integer: 0
sum = 3
```

EXAMPLE: KOTLIN DO...WHILE LOOP

```
fun main() {
    var x:Int = 0
    do {
        x = x + 10
        println("I am inside Do block---"+x)
    } while(x <= 50)
}
```

The output is:

```
I am inside Do block---10
I am inside Do block---20
I am inside Do block---30
I am inside Do block---40
I am inside Do block---50
I am inside Do block---60
```

EXAMPLE: KOTLIN DO...WHILE LOOP

```
fun main(){

    var num = 100
    do {
        println("Loop: $num")
        num++
    }
    while (false)
}
```

The output is:

```
Loop: 100
```

INFINITE DO WHILE LOOP IN KOTLIN

- A do while loop that runs infinitely and never stops is called infinite do while loop.

Example 1:

```
fun main(){
    var num = 100
    do {
        println("Loop: $num")
        num++
    }
    while (true)
}
```

Example 2:

```
fun main(){
    var num = 100
    do {
        println("Loop: $num")
        num--
    }
    while (num<=105)
}
```

Example 3:

```
fun main(){
    var num = 105
    do {
        println("Loop: $num")
        num++
    }
    while (num>=100)
}
```

EXTRA QUESTIONS

```
fun main ()
{
    var right_digit:Int
    println("Enter your number.")
    var number=readLine()!!.toInt()
    while ( number != 0 )
    {
        right_digit = number % 10;
        print("$right_digit");
        number = number / 10;
    }
}
```

The output is:

```
Enter your number.
13579
97531
```

```
fun main ()
{
    var right_digit:Int
    println("Enter your number.")
    var number=readLine()!!.toInt()
    do
    {
        right_digit = number % 10;
        print("$right_digit");
        number = number / 10;
    }while( number != 0 )
}
```

The output is:

```
Enter your number.
13579
97531
```

EXTRA QUESTIONS

```
fun main() {
    var i: Int
    var j: Int
    print("i   j\n")
    print("--  --\n")
    for (i in 1..3) {
        for (j in 1..4)
            print("$i  $j\n")
    }
}
```

The output is:

```
i   j
--  --
1  1
1  2
1  3
1  4
2  1
2  2
2  3
2  4
3  1
3  2
3  3
3  4
```

EXTRA QUESTIONS

```
fun main()
{
    var i=1
    var j=1
    for(i in 1..4)
    {
        for(j in 1..4)
        {
            print("$j");
        }
        print("\n");
    }
}
```

The output is:

```
1234
1234
1234
1234
```

EXTRA QUESTIONS

```
fun main()
{
    var row:Int
    var col:Int
    val n = 6
    for (row in 1..n-1)
    {
        for (col in 1..n-1)
        {
            if (col != 3)
                print("#");
            if (col == 3)
                print("*");
        }
        print("\n");
    }
}
```

The output is:

```
##*##
##*##
##*##
##*##
##*##
```