CM 2062 - Statistical Computing with R
Lab Sheet 9

While Loop in R

In R programming, while loops are used to loop until a specific condition is met.

Syntax of while loop:

```
while (test_expression)
{
statement
}
```

Here, test_expression is evaluated and the body of the loop is entered if the result is TRUE. The statements inside the loop are executed and the flow returns to evaluate the test_expression again. This is repeated each time until test_expression evaluates to FALSE, in which case, the loop exits.

Example

```
 i <- 1
while (i < 6) {
  print(i)
  i = i+1
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

In the above example, i is initially initialized to 1. Here, the test_expression is $i < 6$ which evaluates to TRUE since 1 is less than 6. So, the body of the loop is entered and i is printed and incremented. Incrementing i is important as this will eventually meet the exit condition. Failing to do so will result into an infinite loop. In the next iteration, the value of i is 2 and the loop continues. This will continue until i takes the value 6. The condition $6 < 6$ will give FALSE and the while loop finally exits.

Exercise 1
Write an R programme to print odd numbers which are less than 10.

Exercise 2
Write a R program to calculate the sum of the first ten natural numbers.

Exercise 3
Write a R programme to get factorial of 5 (5!)

break statement with while loop in R

With the break statement, we can stop the loop even if the while condition is TRUE.

Example
In the previous example exit the loop if i is equal to 4.

```
 i <- 1
while (i < 6) {
  print(i)
  i <- i + 1
  if (i == 4) {
    break
  }
}
```

```
[1] 1
[1] 2
[1] 3
```

The loop will stop at 3 because we have chosen to finish the loop by using the break statement when i is equal to 4 (i == 4).

Exercise
Let's take i=1:5 and write and R programme to display the values of i+1 only up to i=3.

next statement with while loop in R

With the next statement, we can skip an iteration without terminating the loop.

Example
Skip the value of 3.

```
 i <- 0
while (i < 5) {
  i <- i + 1
  if (i == 3) {
    next
```

```
  }
  print(i)
}
```

```
[1] 1
[1] 2
[1] 4
[1] 5
```

```
#Second Method
i <- 1
while (i <= 5) {
  if (i == 3) {
    i <- i + 1
    next
  }
  print(i)
  i <- i + 1
}
```

```
[1] 1
[1] 2
[1] 4
[1] 5
```

When the loop passes the value 3, it will skip it and continue to loop.

Exercise

Let's take i=1:5. Write and R programme to display the values of i+1 and skip the value for i=3.

if ... else statement with a while Loop in R

To demonstrate a practical example, let us say we play a game of Yahtzee!

Example

Print "Yahtzee!" If the dice number is 6.

```
 dice <- 1
while (dice <= 6) {
  if (dice < 6) {
    print("No Yahtzee")
  } else {
    print("Yahtzee!")
  }
  dice <- dice + 1
}
```

```
[1] "No Yahtzee"
[1] "No Yahtzee"
[1] "No Yahtzee"
[1] "No Yahtzee"
[1] "No Yahtzee"
[1] "Yahtzee!
```

Nesting for-Loop in while-Loop

It is also possible to nest different types of loops. The below example explains how to nest a for-loop into a while-loop.

```
i <- 1
while(i <= 3) {
  for(j in 1:5) {
    print(paste("This is iteration i =", i, "and j =", j))
  }
  i <- i + 1
}
```

```
[1] "This is iteration i = 1 and j = 1"
[1] "This is iteration i = 1 and j = 2"
[1] "This is iteration i = 1 and j = 3"
[1] "This is iteration i = 1 and j = 4"
[1] "This is iteration i = 1 and j = 5"
[1] "This is iteration i = 2 and j = 1"
[1] "This is iteration i = 2 and j = 2"
[1] "This is iteration i = 2 and j = 3"
[1] "This is iteration i = 2 and j = 4"
[1] "This is iteration i = 2 and j = 5"
[1] "This is iteration i = 3 and j = 1"
[1] "This is iteration i = 3 and j = 2"
[1] "This is iteration i = 3 and j = 3"
[1] "This is iteration i = 3 and j = 4"
[1] "This is iteration i = 3 and j = 5
```

Exercise
Do the above example using nested for loop.

Running while-Loop Through Data Frame Columns

Example

For this example, we are using the iris flower data frame. Let's assume that we want to iterate through these variables until we reach the first non-numeric variable. Until then, we want to add +50 to the values of the variable used in the current iteration.

```
i <- 1
while(is.numeric(iris[ , i])) {
  iris[ , i] <- iris[ , i] + 50
  i <- i + 1
}
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1         55.1        53.5         51.4        50.2  setosa
2         54.9        53.0         51.4        50.2  setosa
3         54.7        53.2         51.3        50.2  setosa
4         54.6        53.1         51.5        50.2  setosa
5         55.0        53.6         51.4        50.2  setosa
6         55.4        53.9         51.7        50.4  setosa
```

```
i <- 1
while(is.numeric(iris[ , i])) {
  if(grepl("Width", colnames(iris)[i])) {
    iris[ , i] <- iris[ , i] + 50
  }
  i <- i + 1
}
head(iris)
```

```
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1         55.1       103.5         51.4       100.2  setosa
2         54.9       103.0         51.4       100.2  setosa
3         54.7       103.2         51.3       100.2  setosa
4         54.6       103.1         51.5       100.2  setosa
5         55.0       103.6         51.4       100.2  setosa
6         55.4       103.9         51.7       100.4  setosa
```

Home Work

Try to get the same output for the examples and exercises that we discuss under for loop using while loop.

Repeat Loop in R

A repeat loop is used to iterate over a block of code multiple number of times. There is no condition check in repeat loop to exit the loop. We must ourselves put a condition explicitly inside the body of the loop and use the break statement to exit the loop. Failing to do so will result into an infinite loop.

Syntax of repeat loop

```
repeat {
   commands
   if(condition) {
      break
   }
}
```

Example

```
x <- 1
repeat {
  print(x)
  x = x+1
  if (x == 6){
    break
  }
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

Exercise 1
Write a R programme to print the text "Hello World" 5 times.

Exercise 2
Write a R programme to print the values from 6 to 10.

Exercise 3
Write a R programme to dispaly the even numbers between 1 to 10.

Running repeat-Loop Over Data Frame Columns

Example 1
Let's consider the "iris" data set. We will use repeat loop to add 100 for each numerical variable.

```
i <- 1
repeat{
  if(!is.numeric(iris[ , i])) {
    break
  }
  iris[ , i] <- iris[ , i] + 100
  i <- i + 1
}
head(iris)
```

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1        105.1       103.5        101.4       100.2  setosa
2        104.9       103.0        101.4       100.2  setosa
3        104.7       103.2        101.3       100.2  setosa
4        104.6       103.1        101.5       100.2  setosa
5        105.0       103.6        101.4       100.2  setosa
6        105.4       103.9        101.7       100.4  setosa
```

Exercise
Write and R programme to add 50 for each width variable in iris data set.

next statement with repeat loop

Example
Let's display values between 1 to 5 except 3 using repeat loop in R.

```
i <- 0
repeat {
  i=i+1
  if (i == 3){
    next
  }
  print(i)
  if (i >= 5){
    break
  }
}
```

```
[1] 1
[1] 2
[1] 4
[1] 5
```

Hypothesis Testing in R

A statistical hypothesis is an assumption made by the researcher about the data of the population collected for any experiment. It is not mandatory for this assumption to be true every time. Hypothesis testing, in a way, is a formal process of validating the hypothesis made by the researcher.

In order to validate a hypothesis, it will consider the entire population into account. However, this is not possible practically. Thus, to validate a hypothesis, it will use random samples from a population. On the basis of the result from testing over the sample data, it either selects or rejects the hypothesis.

Null Hypothesis – The null hypothesis would be considered valid if the alternative hypothesis is fallacious. The evidence that is present in the trial is basically the data and the statistical computations that accompany it. The null hypothesis is denoted by H_0.

Alternative Hypothesis – Hypothesis testing is carried out in order to test the validity of a claim or assumption that is made about the larger population. This claim that involves attributes to the trial is known as the Null Hypothesis. The alternative hypothesis is denoted by H_1 or H_a.

Statisticians use hypothesis testing to formally check whether the hypothesis is accepted or rejected. Hypothesis testing is conducted in the following manner:

1. State the Hypotheses – Stating the null and alternative hypotheses.

2. Formulate an Analysis Plan – The formulation of an analysis plan is a crucial step in this stage.

3. Analyze Sample Data – Calculation and interpretation of the test statistic, as described in the analysis plan.

4. Interpret Results – Application of the decision rule described in the analysis plan.

Hypothesis testing ultimately uses a p-value to weigh the strength of the evidence or in other words what the data are about the population. The p-value ranges between 0 and 1. It can be interpreted in the following way:

A small p-value (≤ 0.05) indicates strong evidence against the null hypothesis, so you reject it.

A large p-value (> 0.05) indicates weak evidence against the null hypothesis, so you fail to reject it.

We will conduct the following hypothesis testing in R.

1. One Sample t-test
2. Two Sample t-test
3. Paired t-test
4. Binomial Test
5. One- Sample Proportion Test
6. Two Samples Proportion Test
7. Chi-square test for interaction