Day 3 and 4 R basic workbook

Chandan Kumar Pandey

08-10-2022

In today's we will discuss the conditional statements. [ex: if -else],loops,and functions [local] in R

Conditional statements

Conditional statements or expression in any computer language will help in making decision. For example if we need to check if the student have passed or fail. Even more, if we are deciding on the grade of the student based on score.

Lets check this example with the case study

```
Student_performance <- read.csv("StudentsPerformance.csv",header = T)
## The data set contain result of 1000 students
## let us see some basic attribute of this data frame
head(Student_performance,n = 10) ## first 10 rows
```

```
##    gender race.ethnicity parental.level.of.education        lunch
## 1  female        group B           bachelor's degree     standard
## 2  female        group C                some college     standard
## 3  female        group B             master's degree     standard
## 4    male        group A          associate's degree free/reduced
## 5    male        group C                some college     standard
## 6  female        group B          associate's degree     standard
## 7  female        group B                some college     standard
## 8    male        group B                some college free/reduced
## 9    male        group D                 high school free/reduced
## 10 female        group B                 high school free/reduced
##    test.preparation.course math.score reading.score writing.score
## 1                     none         72            72            74
## 2                completed         69            90            88
## 3                     none         90            95            93
## 4                     none         47            57            44
## 5                     none         76            78            75
## 6                     none         71            83            78
## 7                completed         88            95            92
## 8                     none         40            43            39
## 9                completed         64            64            67
## 10                    none         38            60            50
```

```
tail(Student_performance,n=10) ##last 10 rows
```

```
##      gender race.ethnicity parental.level.of.education        lunch
## 991    male        group E                high school free/reduced
## 992  female        group B             some high school     standard
## 993  female        group D          associate's degree free/reduced
## 994  female        group D           bachelor's degree free/reduced
## 995    male        group A                high school     standard
## 996  female        group E              master's degree     standard
## 997    male        group C                high school free/reduced
## 998  female        group C                high school free/reduced
## 999  female        group D               some college     standard
## 1000 female        group D               some college free/reduced
##      test.preparation.course math.score reading.score writing.score
## 991                completed         86            81            75
## 992                completed         65            82            78
## 993                     none         55            76            76
## 994                     none         62            72            74
## 995                     none         63            63            62
## 996                completed         88            99            95
## 997                     none         62            55            55
## 998                completed         59            71            65
## 999                completed         68            78            77
## 1000                    none         77            86            86
```

```
str(Student_performance) ##structure of your dataset
```

```
## 'data.frame':    1000 obs. of  8 variables:
##  $ gender                     : chr  "female" "female" "female" "male" ...
##  $ race.ethnicity             : chr  "group B" "group C" "group B" "group A" ...
##  $ parental.level.of.education: chr  "bachelor's degree" "some college" "master's degree" "associate
##  $ lunch                      : chr  "standard" "standard" "standard" "free/reduced" ...
##  $ test.preparation.course    : chr  "none" "completed" "none" "none" ...
##  $ math.score                 : int  72 69 90 47 76 71 88 40 64 38 ...
##  $ reading.score              : int  72 90 95 57 78 83 95 43 64 60 ...
##  $ writing.score              : int  74 88 93 44 75 78 92 39 67 50 ...
```

```
##summary of my data frame
summary(Student_performance)
```

```
##     gender          race.ethnicity     parental.level.of.education
##  Length:1000        Length:1000        Length:1000
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##     lunch           test.preparation.course   math.score     reading.score
##  Length:1000        Length:1000             Min.   :  0.00   Min.   : 17.00
##  Class :character   Class :character        1st Qu.: 57.00   1st Qu.: 59.00
##  Mode  :character   Mode  :character        Median : 66.00   Median : 70.00
##                                             Mean   : 66.09   Mean   : 69.17
##                                             3rd Qu.: 77.00   3rd Qu.: 79.00
##                                             Max.   :100.00   Max.   :100.00
##  writing.score
```

```
##  Min.   : 10.00
##  1st Qu.: 57.75
##  Median : 69.00
##  Mean   : 68.05
##  3rd Qu.: 79.00
##  Max.   :100.00
```

```
##name of the col
names(Student_performance)
```

```
## [1] "gender"                       "race.ethnicity"
## [3] "parental.level.of.education" "lunch"
## [5] "test.preparation.course"     "math.score"
## [7] "reading.score"                "writing.score"
```

```
##in order access an element or col of data frame
#Student_performance$gender
```

If statement in R

Now as you can see from the above output that some student have completed the preparation course while other have not. In order to check if the student have completed the test preparation we will use if statement

if(conditon){function/command}.

```
if(Student_performance$test.preparation.course[1]=="completed"){
  print("I came ready to take the test")
} ## note there is no output here because the first student have not competed this. check the #table ab
if(Student_performance$test.preparation.course[2]=="completed"){
  print("I came ready to take the test")
} ## now you will see the output as second student have completed it.
```

```
## [1] "I came ready to take the test"
```

ifesle statments

Now, using if statement will only execute the command if and only if the statement inside the parenthesis is true. However, in case where the statement is false not output was generate. Now if we want to generate the output in the case were statement is true or false then we have to used if-else statement.

Explaining with the same examples. #if(condition){command/function}else{command}

```
if(Student_performance$test.preparation.course[1]=="completed"){
  print("I came ready to take the test")
}else{
  print("Thats not fare I was not ready")
}
```

```
## [1] "Thats not fare I was not ready"
```

```
if(Student_performance$test.preparation.course[2]=="completed"){
  print("I came ready to take the test")
} else{
  print("Thats not fare I was not ready")
}
```

```
## [1] "I came ready to take the test"
```

In many occasions, we need to take multiple levels of decision. For example while grading. Let us assume that score between 80-100 get A, 60-79 get B and 40-59 get C and below 40 get F

Lets code for this.

```
#for student 8
if(Student_performance$writing.score[8]<40){
  print("You failed, better luck for next exam")
} else if (Student_performance$writing.score[8]<=59 & Student_performance$writing.score[8]>=40){
  print("Your grade is C; need to improve")
} else if (Student_performance$writing.score[8]<=79 & Student_performance$writing.score[8]>=60){
  print("your grade is B; almost there, keep trying")
} else{
  print("Your grade is A; great job")
}
```

```
## [1] "You failed, better luck for next exam"
```

```
# for student 82
if(Student_performance$writing.score[82]<40){
  print("You failed, better luck for next exam")
} else if (Student_performance$writing.score[82]<=59 & Student_performance$writing.score[82]>=40){
  print("Your grade is C; need to improve")
} else if (Student_performance$writing.score[82]<=79 & Student_performance$writing.score[82]>=60){
  print("your grade is B; almost there, keep trying")
} else{
  print("Your grade is A; great job")
}
```

```
## [1] "Your grade is C; need to improve"
```

```
#for student 29
if(Student_performance$writing.score[29]<40){
  print("You failed, better luck for next exam")
} else if (Student_performance$writing.score[29]<=59 & Student_performance$writing.score[29]>=40){
  print("Your grade is C; need to improve")
} else if (Student_performance$writing.score[29]<=79 & Student_performance$writing.score[29]>=60){
  print("your grade is B; almost there, keep trying")
} else{
  print("Your grade is A; great job")
}
```

```
## [1] "your grade is B; almost there, keep trying"
```

```r
# for student number 3
if(Student_performance$writing.score[3]<40){
  print("You failed, better luck for next exam")
} else if (Student_performance$writing.score[3]<=59 & Student_performance$writing.score[3]>=40){
  print("Your grade is C; need to improve")
} else if (Student_performance$writing.score[3]<=79 & Student_performance$writing.score[3]>=60){
  print("your grade is B; almost there, keep trying")
} else{
  print("Your grade is A; great job")
}
```

```
## [1] "Your grade is A; great job"
```

```r
Student_performance[c(8,82,29,3),]
```

```
##    gender race.ethnicity parental.level.of.education        lunch
## 8    male        group B                some college free/reduced
## 82   male        group B                 high school free/reduced
## 29   male        group C                 high school     standard
## 3  female        group B             master's degree     standard
##    test.preparation.course math.score reading.score writing.score
## 8                     none         40            43            39
## 82                    none         49            45            45
## 29                    none         70            70            65
## 3                     none         90            95            93
```

Some time it is within a if statement there is another if else statement. Such scenarios is called nested is else statement. For example let us assume that passing mark for ethnic group B is 35 while group A,c and D is 41. Let understand this example by code.

```r
## for student number 8 in maths
if(Student_performance$race.ethnicity[116]=="group B"){
  if(Student_performance$math.score[116]>=35){
    print("You are group B and Pass")
  } else { print("Fail")
    }
}else {
  if(Student_performance$math.score[116]>40){
    print("You are not from group B and Pass")
  } else {print("Fail")
    }
}
```

```
## [1] "You are not from group B and Pass"
```

Looping the command

You can clearly see from the above examples, we have to apply same logic for all rows of data frame. It is a repetitive process. so we can run the same script in the loop.

For example, to pass the exam the student need to

- score min of 40 in each paper.
- get on average 45 in all three paper.

Now this task require applying the same logic for all rows.

For loop or definte loop

```
##creating a vector with all NA
final_results <- rep(NA,nrow(Student_performance))
for(i in 1:nrow(Student_performance)){
  if(Student_performance$math.score[i]>=40 &
     Student_performance$reading.score[i]>=40 &
     Student_performance$writing.score[i]>=40){
    average_percent<- sum(Student_performance[i,6:8])/3
    if(average_percent>=45){
      final_results[i]=round(average_percent,2)
    }else{
      final_results[i]="Fail"
    }
  }else{
    final_results[i]="Fail"
  }
}
Student_performance$final_result <- final_results
```

For example 2

In your first example we have looped over the increase of i , then i was used to amend the values in the vector, *final_result*. In this case the value of i starts with 1 and increase by 1 till it reaches 1000.

However one can directly loop over the name of the vector. Let see with the example. If we have given gender code F to female and M to male

```
t=1
Gender_code<-rep(NA,nrow(Student_performance))
for(j in Student_performance$gender){
  if(j=="male"){
    Gender_code[t]="M"
  }else{
    Gender_code[t]="F"
  }
  t=t+1
}
Student_performance$gender_code<-Gender_code
```

while loop or indefente loop

In both cases above we were sure that now many time loop will iterate. However, some time we do not have idea how many time the loop will iterate. This is called indefinite loop or while loop. Let say we have to generate a random number and add to existing number till the number is greater that equal to 200

```
My_number <- 0 #stating with zero
flag <- 1
while(My_number<=200){
##generating a random number between 1,10
rand_number <- sample(1:10,1,replace = T)
##adding the random sampled number to my number
My_number <- My_number+rand_number
flag <- flag + 1
}
print(flag)
```

```
## [1] 38
```

Function

Sometime you have to perform same series of calculation with all the time If you have to create pass or grade table every year or for many classes. In these cases it is helpful to write your own function. Let us check if the student have passed or not.

```
grade_of_student<-function(marks){
  final_results <- rep(NA,nrow(marks))
for(i in 1:nrow(marks)){
  if(marks$math.score[i]>=40 &
     marks$reading.score[i]>=40 &
     marks$writing.score[i]>=40){
    average_percent<- sum(marks[i,6:8])/3
    if(average_percent>=45){
      final_results[i]=round(average_percent,2)
    }else{
      final_results[i]="Fail"
    }
  }else{
    final_results[i]="Fail"
  }
}
  return(final_results)
}
Student_performance$grade_fun <- grade_of_student(Student_performance)
```

frequent used command in R

1. length() : length() of the vector or list.

```
x <- 1:50
length(x)
```

```
## [1] 50
```

2. subset() : subset the data frame as per condition.

```
fail_students <- subset(Student_performance,Student_performance$final_result=="Fail")
head(fail_students)
```

```
##    gender race.ethnicity parental.level.of.education        lunch
## 8    male        group B                some college free/reduced
## 10 female        group B                 high school free/reduced
## 18 female        group B            some high school free/reduced
## 19   male        group C             master's degree free/reduced
## 34   male        group D                some college     standard
## 56 female        group C                 high school free/reduced
##    test.preparation.course math.score reading.score writing.score final_result
## 8                     none         40            43            39         Fail
## 10                    none         38            60            50         Fail
## 18                    none         18            32            28         Fail
## 19               completed         46            42            46         Fail
## 34                    none         40            42            38         Fail
## 56                    none         33            41            43         Fail
##    gender_code grade_fun
## 8            M      Fail
## 10           F      Fail
## 18           F      Fail
## 19           M      Fail
## 34           M      Fail
## 56           F      Fail
```

3. which() : Give the TRUE indices of a logical object, allowing for array indices.

```
pass_index <- which(Student_performance$final_result!="Fail")

pass_student <-  Student_performance[pass_index,]
head(pass_student,10)
```

```
##    gender race.ethnicity parental.level.of.education        lunch
## 1  female        group B           bachelor's degree     standard
## 2  female        group C                some college     standard
## 3  female        group B             master's degree     standard
## 4    male        group A          associate's degree free/reduced
## 5    male        group C                some college     standard
## 6  female        group B          associate's degree     standard
## 7  female        group B                some college     standard
## 9    male        group D                 high school free/reduced
## 11   male        group C          associate's degree     standard
## 12   male        group D          associate's degree     standard
##    test.preparation.course math.score reading.score writing.score final_result
## 1                     none         72            72            74        72.67
## 2                completed         69            90            88        82.33
## 3                     none         90            95            93        92.67
## 4                     none         47            57            44        49.33
## 5                     none         76            78            75        76.33
## 6                     none         71            83            78        77.33
## 7                completed         88            95            92        91.67
## 9                completed         64            64            67           65
```

```
## 11                    none         58            54            52        54.67
## 12                    none         40            52            43           45
##    gender_code grade_fun
## 1            F     72.67
## 2            F     82.33
## 3            F     92.67
## 4            M     49.33
## 5            M     76.33
## 6            F     77.33
## 7            F     91.67
## 9            M        65
## 11           M     54.67
## 12           M        45
```

4. sort() : sorting the vector
5. apply() : apply function. There are many type of apply functions. Some example are *sapply() and tapply*

```
sapply(Student_performance[,6:8], mean) ## mean of each subject
```

```
##    math.score reading.score writing.score
##        66.089        69.169        68.054
```

```
tapply(Student_performance$math.score, Student_performance$gender, mean) ## mean base on unique #value
```

```
##   female     male
## 63.63320 68.72822
```