|  |
| --- |
| Solution Provider Name: Mrunali Mahesh Sawant (MCA Student)  College: V.P.Institute of Mangement Studies & Research, Sangli  R Version : R version 3.6.3  R studio : 1.3.1093 |

**Introduction to Decision Making in R Programming**

Decision making is significant aspect in every programming language. To check the conditions in form of TRUE and FALSE values we can get appropriate result with the help of decision making statements. R supports a decision making ability to take a decision in case of providing result by satisfying condition. R provides different decision making or conditional statements like IF, ELSE\_IF.

R language also provides the looping statements for getting result in from particular range by satisfying the condition. The looping statements in R like FOR, WHILE and REPEAT etc. R supports a different operators like operator like logical (&), logical (|), arithmetic operator like (+,\*) and relational operator like (==) etc. that can be used for in case of checking condition in decision making phase.

**Vector:** A vector is a basic data structure used in R programming. A vector stores the same type of element, the types of element is may be an integer, float, string or character. A vector concept is same as “Array” concept in procedural programming language and object oriented programming language. For creating a vector in R, R provides **c () function.**

**Data Frame:**  A data frame is two dimensional structure of data used in R. It is a special case of list which contains equal length of components. It takes components from column and contents from rows. We can create a data data frame suing data. Frame () function provided by R. Combine all vectors in list using data. Frame () function followed by the vectors name separated by commas.

**Read :** A read() method used to read the contents of .csv file in R language. A read() followed by .csv file name in double quotes read the information from csv file.

**Note:** R programming is the case sensitive language so we have to write commands as same as commands provided by R.

For run all commands of script, select commands and press **ctrl+enter.**

**Experiments:**

**Consider the following data set and solve the following problems.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **EmpNo** | **Age** | **Height** | **Weight** | **Basic Salary** | **Experience** | **Location** |
| 101 | 45 | 6.2 | 67 | 10000 | 6.3 | Sangli |
| 102 | 23 | 4.7 | 89 | 6700 | 7.2 | Satara |
| 103 | 26 | 5.2 | 54 | 6400 | 2.5 | Nashik |
| 104 | 31 | 5.5 | 68 | 8400 | 7.5 | Kolhapur |
| 105 | 30 | 5.4 | 44 | 8450 | 4.8 | Jalgao |
| 106 | 22 | 4.9 | 56 | 8100 | 9.4 | Pune |
| 107 | 26 | 6.2 | 51 | 9500 | 3.5 | Jalgao |
| 108 | 24 | 4.6 | 55 | 15000 | 8.4 | Satara |
| 109 | 28 | 5.1 | 69 | 18450 | 9.0 | Sangli |
| 110 | 29 | 5.9 | 89 | 7400 | 4.6 | Satara |
| 111 | 48 | 5.4 | 95 | 25000 | 8.2 | Pune |
| 112 | 42 | 4.3 | 40 | 22000 | 9.4 | Kolhapur |
| 113 | 49 | 4.9 | 77 | 9000 | 1.5 | Nashik |
| 114 | 32 | 5.8 | 70 | 9400 | 1.6 | Jalgoa |
| 115 | 33 | 6.8 | 74 | 12400 | 2.7 | Pune |
| 116 | 46 | 6.2 | 60 | 45000 | 4.8 | Satara |
| 117 | 36 | 4.3 | 48 | 43000 | 8.7 | Pune |
| 118 | 35 | 5.9 | 59 | 17500 | 9.4 | Nashik |
| 119 | 32 | 5.5 | 66 | 18600 | 10.5 | Kolhapur |
| 120 | 27 | 4.5 | 80 | 17800 | 11.4 | Pune |

**Introduction:-**

The purpose of this experiment is to understand the concept of reading contents of .csv file using read command.

**Procedure:**

1. Open R studio take a new script.
2. Write rm(list=ls()) command on script to clear workspace.
3. Read EmpDetails.csv file using read command in R and store it result on Employee\_Information variable.
4. Print Employee\_Information.
5. Run all commands by pressing ctrl+enter.
6. Save the script with name R\_code\_decision extension .r on desired location like Excersice\_1.r

**Code and Result :-**

**Open R studio take a new script.**

#1.open R studio and take new script

#1.1 open R studio and go to file option

#1.2 after selecting file option and click new File and after selecting new file another window is appeared on screen in that select R script.

#1.3 the untitled R script in opened on screen.

**Write rm(list=ls()) command on script to clear workspace**

#2. Write rm(list=ls()) command in script for clearing all workspace.

rm(list=ls())

**Read EmpDetails.csv file using read command in R and store the result of it on Employee\_Information variable**

#3. read EmpDetails.csv file using read command and store it on Employee\_Information variable

Employee\_Information<-read.csv("EmpDetails.csv")

**Print Employee\_Information using print function.**

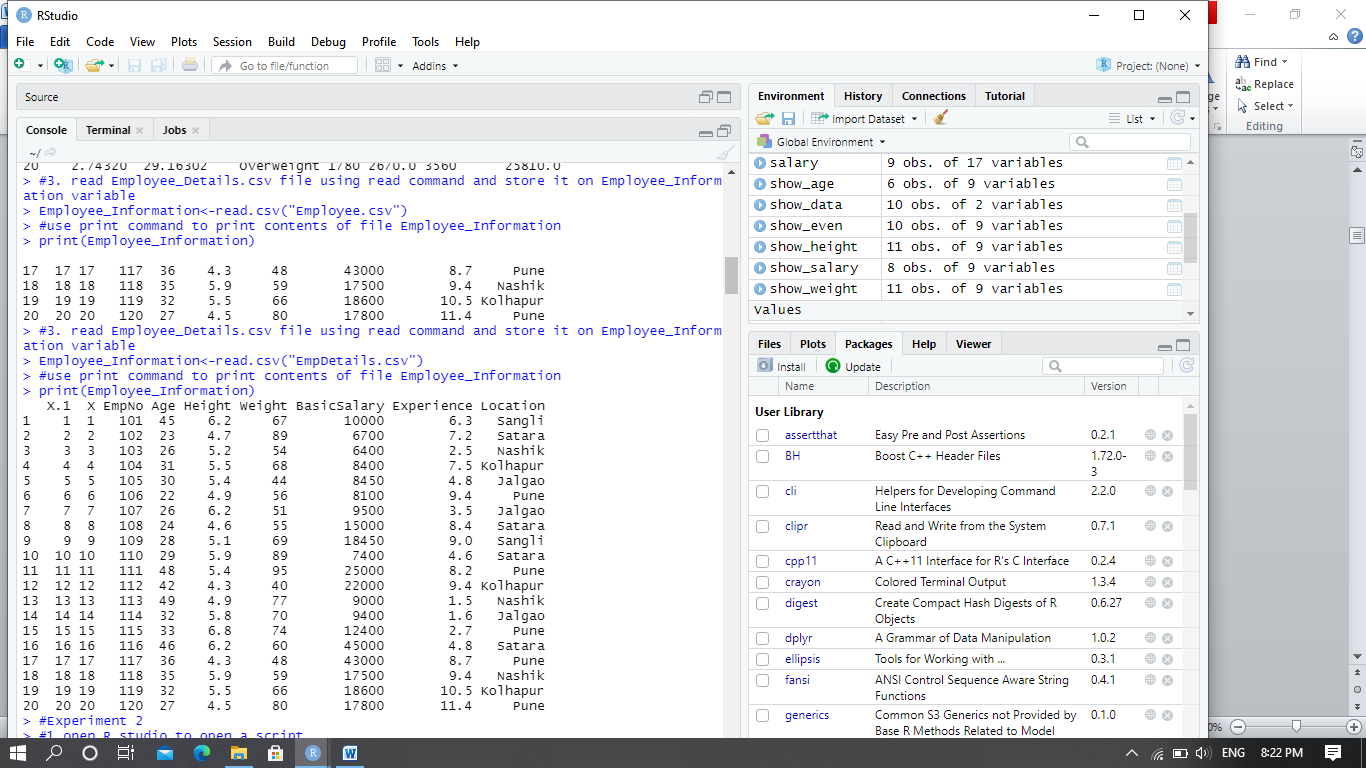
#use print command to print contents of file EmpDetails.csv

print(Employee\_Information)

**Save this script using save option from file.**

Save script Exercise\_1.r using save option.

**Output : -**



**Conclusion: -**

We can read the contents of .csv file in R using read() method in R .

**1. Display employees whose age is between 30 to 35.**

**Introduction:**

The purpose of this experiment is to understand the concept of accessing elements within a range using logical operator &. And use the subset () function for display values which satisfies the condition written in subset () function.

**Procedure:-**

1. Create a variable as show\_age . Create subset with condition Age>=30 & Age<=35
2. Print show\_age
3. Run all commands by pressing ctrl+enter .
4. Save the script using save option from file.

**Code And Result :-**

**Open R studio and open Excersice\_1.r script**

**Create a variable as show\_age . Create subset with condition Age>=30 & Age<=35**

#Create a variable as show\_age . Create subset with condition Age>=30 & Age<=35

show\_age<-data.frame(subset(Employee\_Information,Age>=30 & Age<=35))

**Print show\_age**

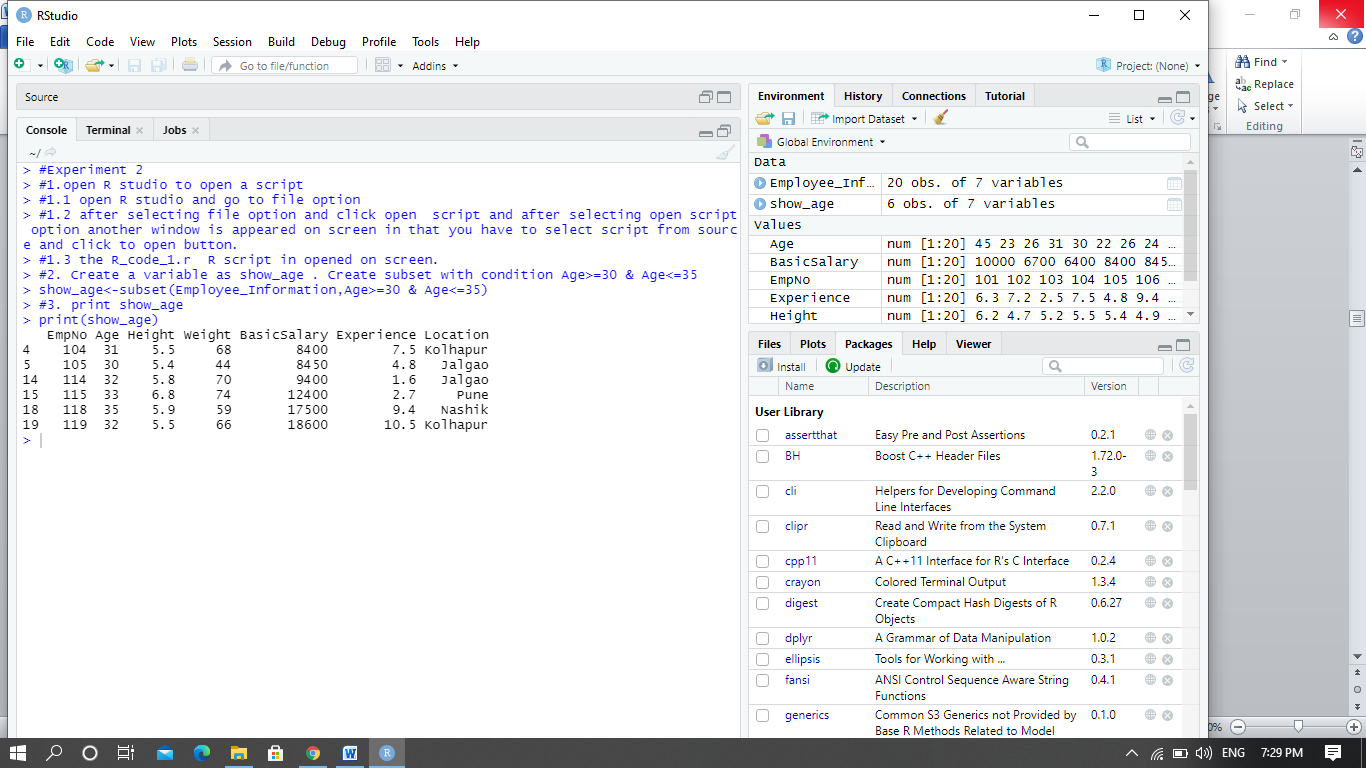
#3. print show\_age

print(show\_age)

**Save the script Excersice\_1.r on using save option..**

Save script with name Exercise\_1.r

**Output –**



**Conclusion –**

We understand the concept of using logical operator & ‘AND’. We can use subset () function to make a subset of given data. We can perform basic arithmetic operations on elements using logical operator and make separate set of those operations result using subset () function.

**2. Display employees whose salary >15000.**

1. A relational > operator used when you want to display values greater than from particular value.
2. We can use subset() function to perform this kind of basic operations to get result in another subset of data.
3. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept of conditional operator ‘>’ greater than for performing operations and store a result in subset.

**Procedure:-**

1. Create a variable as show salary. Create subset with condition BasicSalary>15000
2. Print show\_salary.
3. Run all commands by pressing ctrl+enter.
4. Save the script using save option from file.

**Code And Result :-**

**Create a variable as show salary. Create subset with condition BasicSalary>15000**

#Create a variable as show salary. Create subset with condition BasicSalary>15000

show\_salary <-subset(Employee\_Information,BasicSalary>15000)

**Print show\_salary**

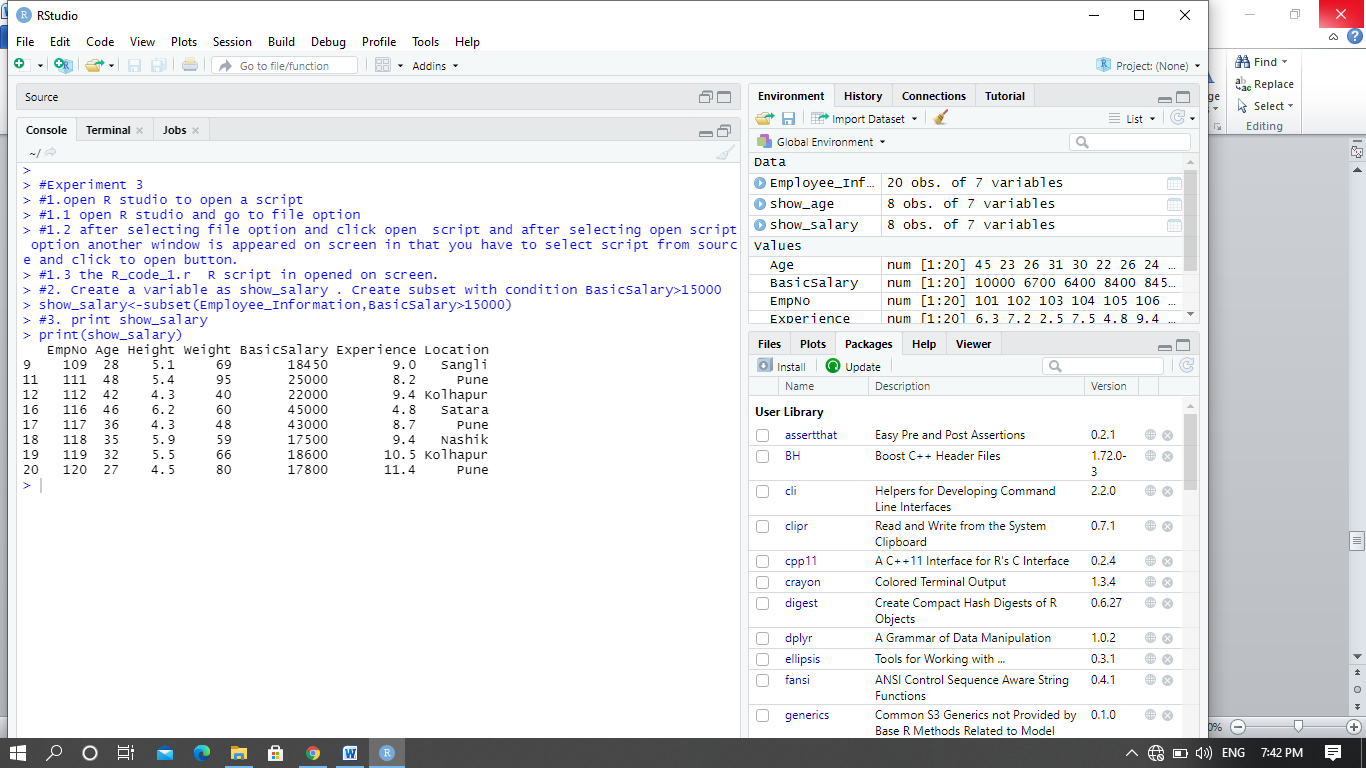
#3. print show\_salary

print (show\_salary)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output –**



**Conclusion –**

We understand the concept of using relational operator > ‘Greater Than’ and storing a result of operation in subset.

**3. Display employees details whose EmpNo is even.**

1. **Arithmetic %% operator: -** A %% operator used when you want to calculate reminder of value.
2. **Relational == operator: -** A ‘==’ exact equals to operator is used when display result when resulted values are exactly equals to expected values.
3. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
4. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept of use of Relational ‘ **%%’ modulus operator** and **‘==’ exact equals to** operator.

**Procedure:-**

1. Open R studio open R\_code\_decision.r script.
2. Create a variable as show even. Create subset with condition for even number on EmpNo element using %% and == operators.
3. Print show even.
4. Run all commands by pressing ctrl+enter.
5. Save the script using save option from file.

**Code & Result:-**

**Open R studio and open Excersice\_1.r script**

**Create a variable as show\_even. Create subset with condition for even number on EmpNo element using %% and == operators.**

#1. Create a variable as show\_even. Create subset with condition for even number on EmpNo element using %% and == operators.

show\_even<-subset (Employee\_Information,EmpNo%%2==0)

**Print show\_even**

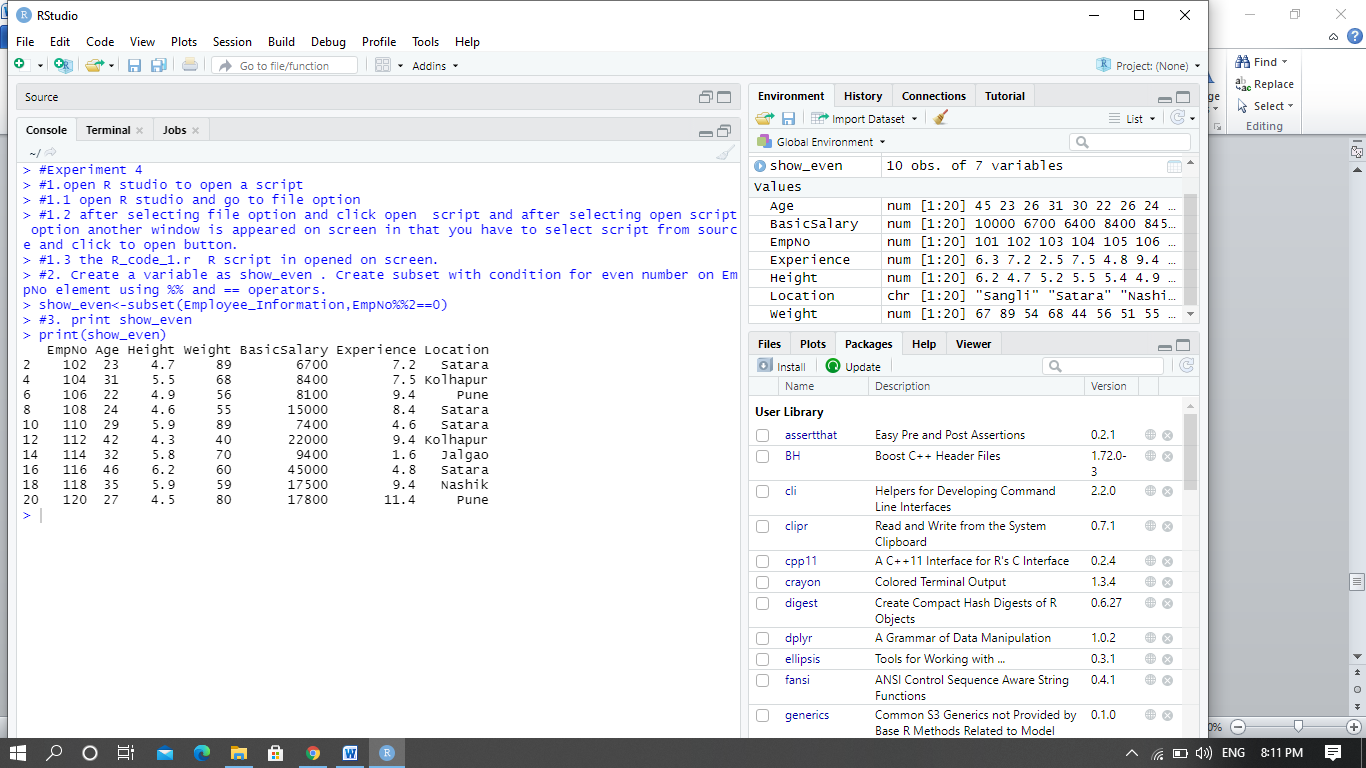
#2. print show\_even

print(show\_even)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output –**



**Conclusion –**

We understand the concept of using arithmetic operator %% ‘Modulus operator’ and relational ‘== ‘exact equals to ’ to performing logic of even numbers on elements .It shows result only when given condition satisfies and storing a result of operation in subset.

**4. Display employees whose height <5.5 and weight>60.**

**How to extract a values of elements by satisfying given conditions using relational operators ?**

1. **Relational > operator: -** A greater than > operator used when you want to display value greater than particular values specified in condition.
2. **Relational < operator: -** A less than > operator used when you want to display value less than particular values specified in condition.
3. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
4. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept of use of Relational operators greater than and less than.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Create a variable as show\_height . Create subset with condition displaying Height<5.5 .
3. Print show\_height.
4. Create a variable as show\_weight . Create subset with condition displaying Weight>60 .
5. Print show\_weight
6. Run all commands by pressing ctrl+enter .
7. Save the script using save option from file.

**Code and Result:-**

**Open R studio and open Excersice\_1.r script**

**Create a variable as show\_height . Create subset with condition displaying Height<5.5 .**

#2. Create a variable as show\_height . Create subset with condition for displaying height<5.5.

show\_height<-subset(Employee\_Information,Height<5.5)

**Print show\_height**

#3. print show\_height

print(show\_height)

**Create a variable as show\_weight . Create subset with condition displaying Weight>60 .**

#4. Create a variable as show\_weight . Create subset with condition for displaying weight>60.

show\_weight<-subset(Employee\_Information,Weight>60)

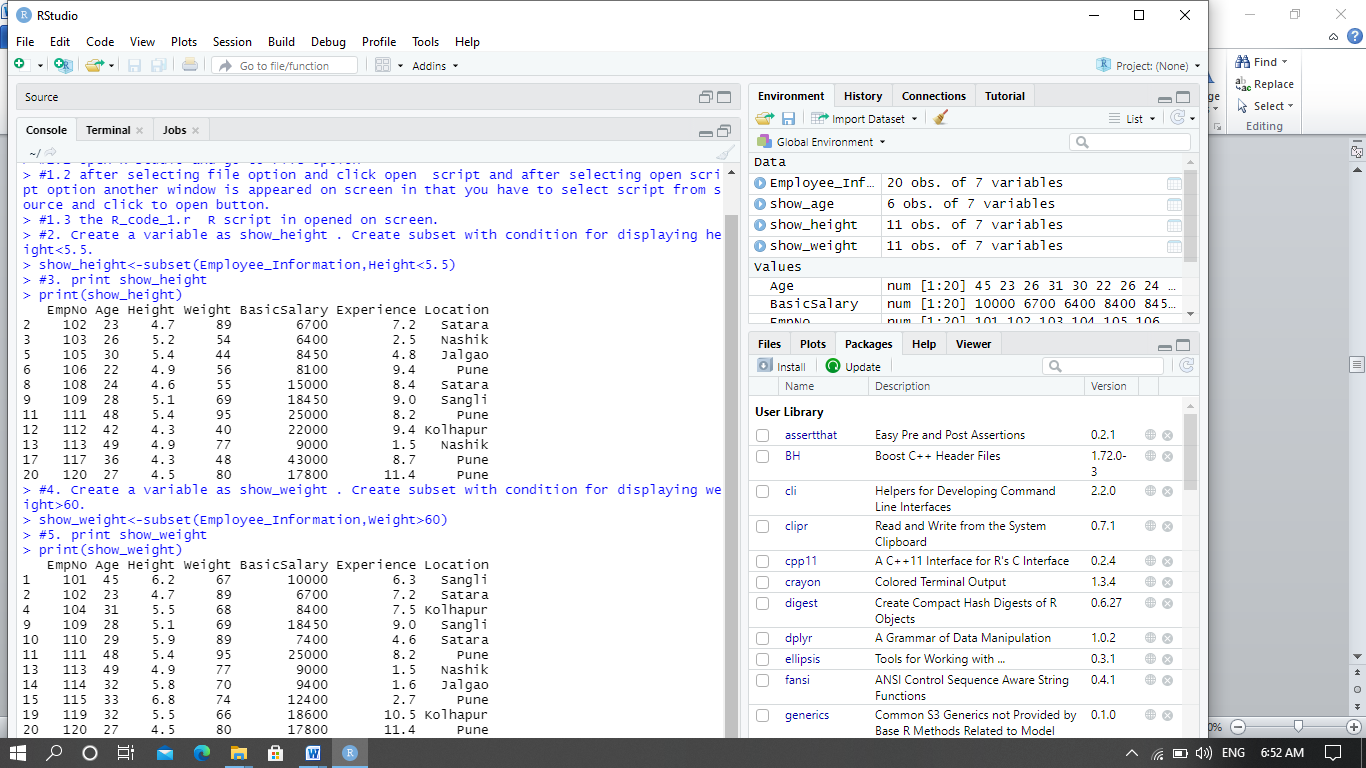
**Print show\_weight**

#5. print show\_weight

print(show\_weight)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output –**

**Conclusion –**

We understand the concept of using relational operators like greater than “>” and less than “<”.It shows result only when given condition satisfies and storing a result of operation in subset.

**5. Check employees height, weight and display whether persons fit, overweight or underweight.**

**Introduction:**

The purpose of this experiment is to understand the concept of creating/ adding a new column/element in data frame and use of decision statements and looping statements in R.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Create a variable temp and assign operation of converting height fit to meters.
3. Create a new element Height\_in\_meter in Employee\_Information and assign temp variable values to Height\_in\_meter.
4. Print Employee\_Information$Height\_in\_meter
5. Create a variable temp1 and assign operation of making square of Height\_in\_meter element.
6. Print temp1
7. Create a new element Height\_sqr in Employee\_Information and assign temp1 variable values to Height\_sqr.
8. Print Employee\_Information$Height\_sqr
9. Create element BMI\_ratio and assign operation of calculating BMI ration using Weight and Height\_sqr element.
10. print Employee\_Information$BMI\_ratio
11. View Employee\_Information$BMI\_ratio
12. In for loop use e1 variable and take range 1: nrow(Employee\_Information) for whole dataset.
13. check condition using IF statement for fit , underweight and overweight
14. Add new columnn as a Result\_of\_BMI and add assign the value as Underweight when the given condition is true.
15. Use else if statements to check whether the element satisfies a second condition when the first condition is false.
16. Assign values as Fit to Result\_of\_BMI column when the given condition is true.
17. Use else to display else part.
18. Assign values as Overweight to Result\_of\_BMI column when the above conditions are false and else part is executed.
19. Print Employee\_Information data frame to check desired result.
20. View Employee\_Information data frame to check desired result in tabular form.
21. Run all commands by pressing ctrl+enter.
22. Save the script using save option from file.

**Code and Result:-**

#1.Create a variable temp and assign operation of converting height fit to meters

temp<-Employee\_Information$Height\*0.3048

#2.Create a new element Height\_in\_meter in Employee\_Information and assign temp variable values to Height\_in\_meter.

Employee\_Information$Height\_in\_meter<-temp

**Print Employee\_Information$Height\_in\_meter**

#print Employee\_Information$Height\_in\_meter

print(Employee\_Information$Height\_in\_meter)

#3.Create a variable temp1 and assign operation of making square of Height\_in\_meter element.

temp1<-Employee\_Information$Height\_in\_meter\*2

**Print temp1**

#print temp1

print(temp1)

#4.Create a new element Height\_sqr in Employee\_Information and assign temp1 variable values to Height\_sqr.

Employee\_Information$Height\_sqr<-temp1

**Print Employee\_Information$Height\_sqr**

#print Employee\_Information$Height\_sqr

print (Employee\_Information$Height\_sqr)

#5. Create element BMI\_ratio and assign operation of calculating BMI ration using Weight and Height\_sqr element.

Employee\_Information$BMI\_ratio<-Employee\_Information$Weight/Employee\_Information$Height\_sqr

**Print Employee\_Information$BMI\_ratio**

#print Employee\_Information$BMI\_ratio

print(Employee\_Information$BMI\_ratio)

**View Employee\_Information$BMI\_ratio**

#View Employee\_Information$BMI\_ratio

View(Employee\_Information$BMI\_ratio)

**In for loop use e1 variable and take range 1:nrow(Employee\_Information) for whole dataset.**

#6.In for loop use e1 variable and take range 1:nrow(Employee\_Information) for whole dataset.

for(e1 in 1:nrow(Employee\_Information))

{

**Check condition using IF statement for fit , underweight and overweight**

#check condition using IF statement for fit , underweight and overweight

if(Employee\_Information$BMI[e1]>18.5)

{

**Add new columnn as a Result\_of\_BMI and add assign the value as Underweight when the given condition is true.**

#Add new columnn as a Result\_of\_BMI and add assign the value as Underweight when the given condition is true.

Employee\_Information$Result\_of\_BMI[e1]<-"UnderWeight"

}

**Use else if statements to check whether the element satisfies a second condition when the first condition is false**

# Use else if statements to check whether the element satisfies a second condition when the first condition is false.

else if(Employee\_Information$BMI[e1]>=18.5 Employee\_Information$BMI[e1]<=24)

{

**Assign values as Fit to Result\_of\_BMI column when the given condition is true.**

#Assign values as Fit to Result\_of\_BMI column when the given condition is true.

Employee\_Information$Result\_of\_BMI[e1]<-"Fit"

}

**Use else to display else part.**

# Use else to display else part.

else

{

**Assign values as Overweight to Result\_of\_BMI column when the above conditions are false and else part is executed.**

#Assign values as Overweight to Result\_of\_BMI column when the above conditions are false and else part is executed.

Employee$Result\_of\_BMI[e1]<-"OverWeight

}

}

**Print Employee\_Information data frame to check desired result.**

#8.Print Employee\_Information data frame to check desired result.

print(Employee\_Information)

**View Employee\_Information data frame to check desired result in tabular form.**

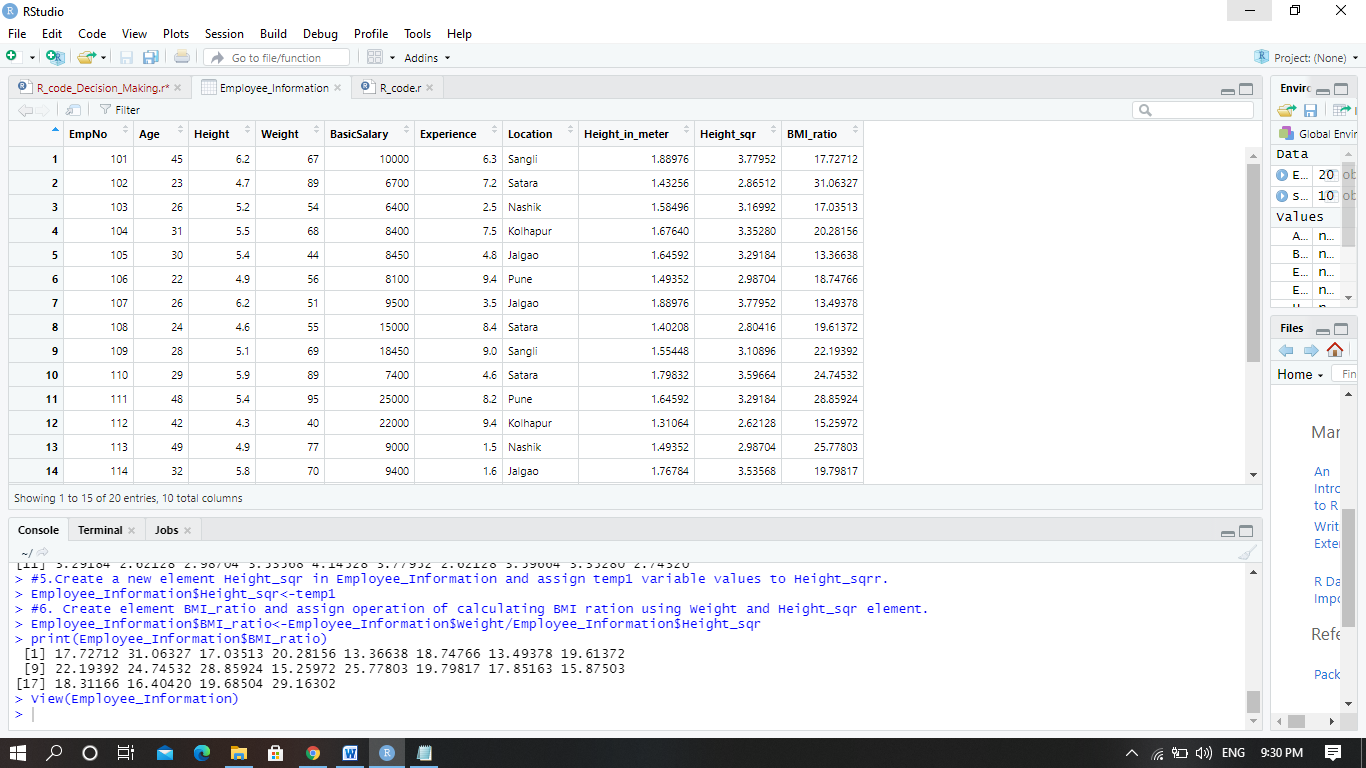
#9.Print Employee\_Information data frame to check desired result.

View(Employee\_Information)

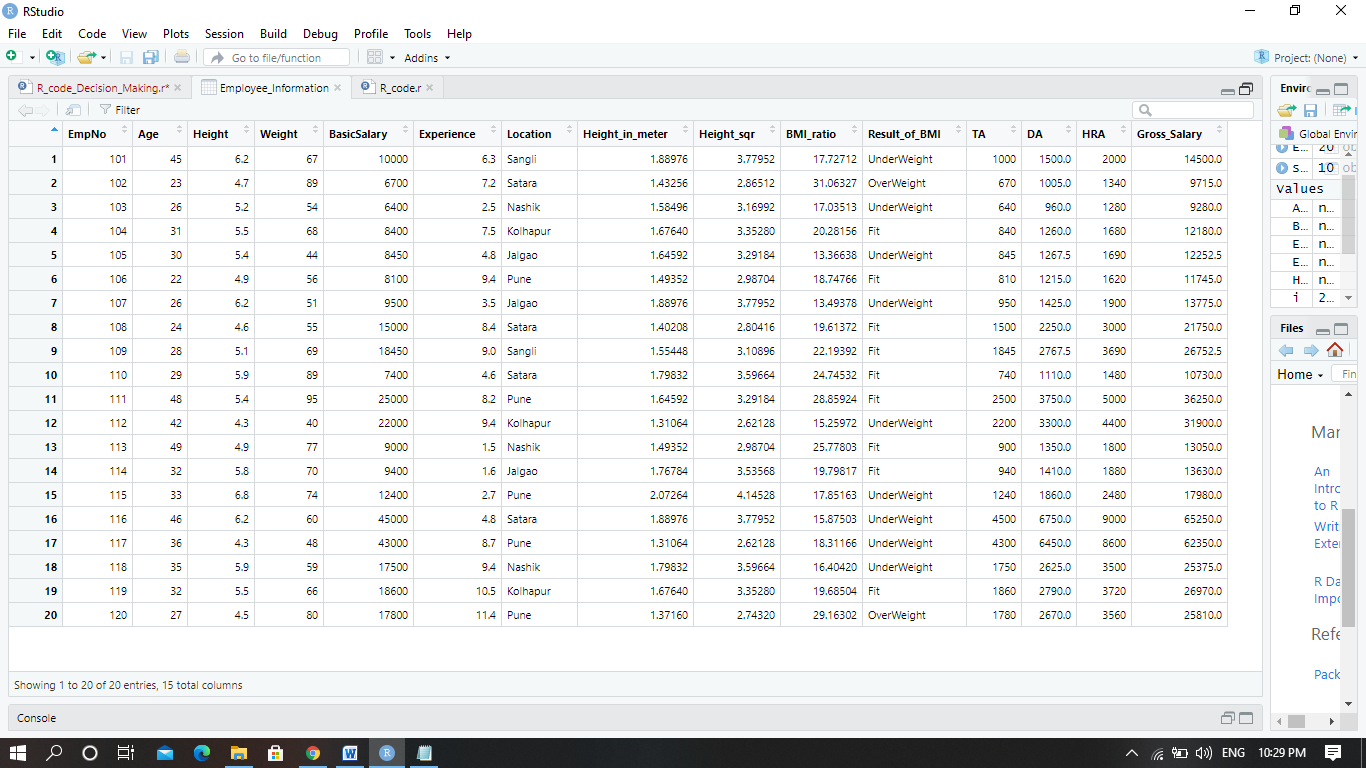
**Save the script Excersise\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output – 1. BMI ratio :**



**2. For Fit, Overweight, and Underweight Condition:**



**Conclusion –**

We understand the concept of decision making statements and looping statements in R. We can add the new column or element of data frame in existing data frame. We can perform different operations in R is easily possible using R contents

.

**6. Calculate gross salary of employees (TA is 10%, DA is 15%, HRA is 20%) on basic salary.**

**How to calculate Gross salary on the basis of given conditions using given dataset?**

1. **Gross Salary: - Gross salary** is mathematical calculation for salary payable to employee after adding different allowances and deducting some taxes from his basic salary.

**Syntax for Gross Salary:**

Gross = BasicSalary-TA+DA+HRA

1. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept to calculate gross salary from given data set.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Add new column as TA add calculate TA 10% on BasicSalary.
3. Add new column as DA add calculate DA 15% on BasicSalary.
4. Add new column as HRA add calculate HRA 20% on BasicSalary.
5. View Employee\_Information
6. Add new column Gross\_Salary and calculate Gross salary and assign the result to this column.
7. View Emp\_saldata data frame to get desired result.
8. Run all commands by pressing ctrl+enter.
9. Save the script using save option from file.

**Code And Result :-**

**Add new column as TA add calculate TA 10% on BasicSalary.**

#1.Add new column as TA add calculate TA 10% on BasicSalary.

Employee\_Information$TA<-Employee\_Information$BasicSalary\*10/100

**Add new column as DA add calculate DA 15% on BasicSalary.**

#2.Add new column as DA add calculate DA 15% on BasicSalary.

Employee\_Information$DA<-Employee\_Information$BasicSalary\*15/100

**Add new column as HRA add calculate HRA 20% on BasicSalary.**

#3.Add new column as HRA add calculate HRA 20% on BasicSalary.

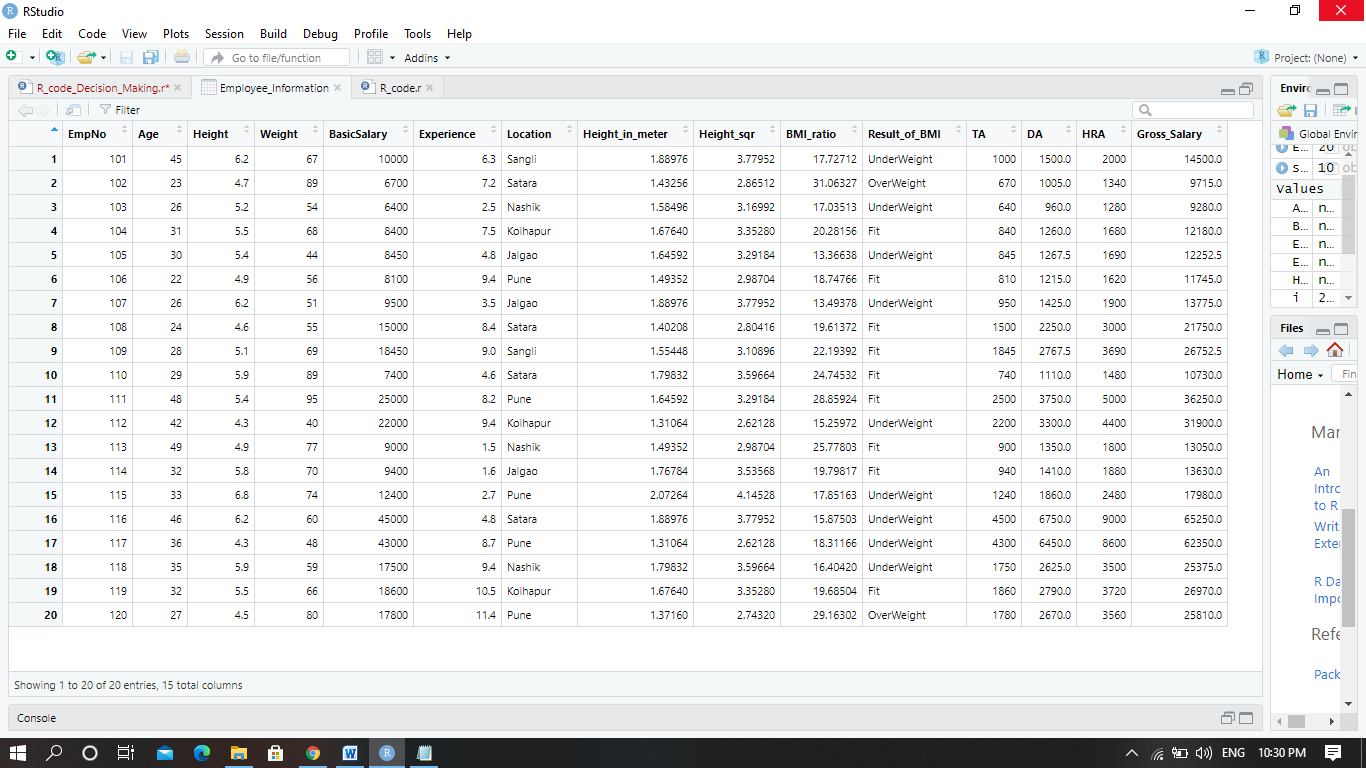
Employee\_Information$HRA<-Employee\_Information$BasicSalary\*20/100

**View Employee\_Information data frame to get desired result.**

#4.View Employee\_Information data frame

View (Employee\_Information)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output -**

**Conclusion –**

We can calculate the different formulas like gross easily using the R contents.

**7. Display employee details whose gross salary is >20000.**

**How to extract data element value from data frame as per the condition using relational greater than operator ‘>’ and how to store a result in newly created subset?**

1. **Relational > operator: -** A logical > operator used when you want to display values greater than from particular value.
2. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
3. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept of how to use relational greater than operator to satisfy the condition and store the result in subset.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Take a new variable as salary and create a subset of Employee\_Information whose Gross\_Salary >20000.
3. Print salary variable to get desire a result
4. Run all commands by pressing ctrl+enter .
5. Save the script using save option from file.

**Code And Result :-**

**Take a new variable as salary and create a subset of Employee\_Information whose Gross\_Salary >20000.**

#Take a new variable as salary and create a subset of Employee\_Information whose Gross\_Salary >20000.

salary<-subset(Employee\_Information,Gross\_Salary>20000)

**Print salary variable to get desire a result**

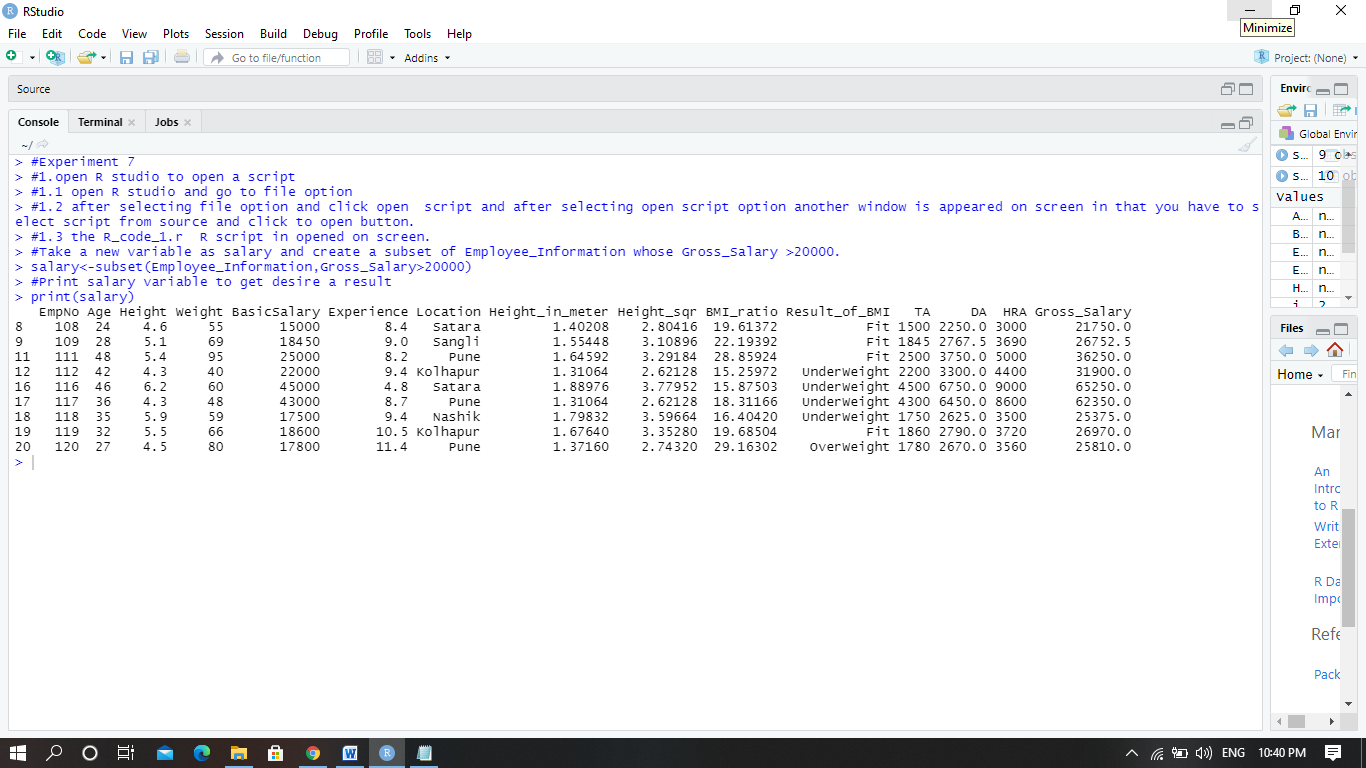
#Print salary variable to get desire a result

print(salary)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output –**



**Conclusion –**

We can extract the values of element by satisfying conditions with help of relational operators and store their result in newly created subset.

**8. Display employee details who’s having more than 5 years’ experience.**

**How to extract data element value from data frame as per the condition using relational greater than operator ‘>’ and how to store a result in newly created subset?**

1. **Relational > operator: -** A relational> operator used when you want to display values greater than from particular value.
2. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
3. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept of how to use relational greater than operator to satisfy the condition and store the result in subset.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Take a new variable as experience and create a subset of Employee\_Information whose Experience>5.
3. Print experience variable to get desire a result
4. Run all commands by pressing ctrl+enter.
5. Save the script using save option from file.

**Code and Result:-**

**Open R studio and open Excersice\_1.r script**

#1.open R studio to open a script

**Take a new variable as experience and create a subset of Employee\_Information whose Experience >5.**

#Take a new variable as experience and create a subset of Employee\_Information whose Experience >5.

experience<-subset(Employee\_Information,Experience>5)

**Print experience variable to get desire a result**

#Print experience variable to get desire a result

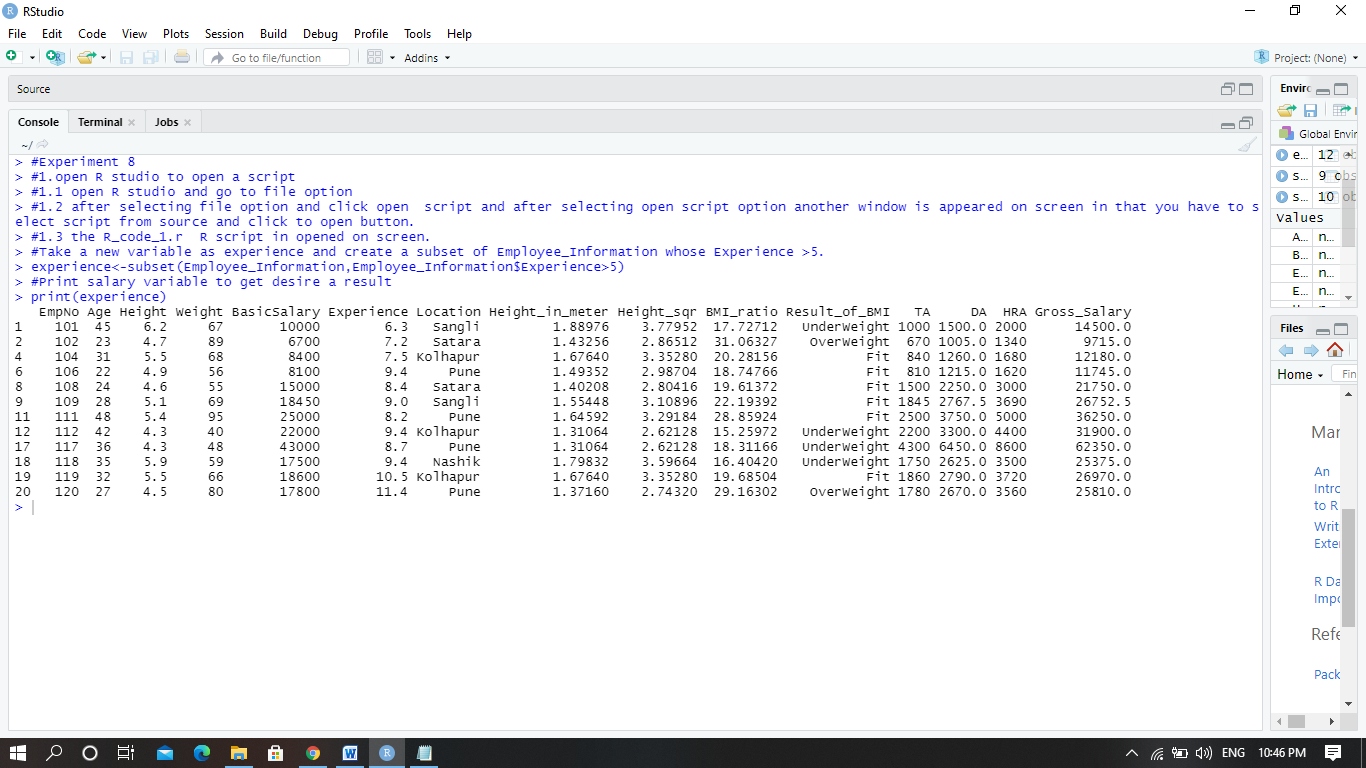
print(experience)

**Save the script Excersice\_1.r on using save option.**

Save script with name Excersice\_1.r

**Output –**

**Conclusion –**

We can extract the values of element by satisfying conditions with help of relational operators and store their result in newly created subset.

**9. Display employee details whose location is pune.**

**How to extract data element value from data frame as per the condition using relational exact equal to operator ‘==’ and how to store a result in newly created subset?**

1. **Relational == operator: -** A ‘==’ exact equals to operator is used when display result when resulted values are exactly equals to expected values.
2. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
3. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept to use of relational operator exact equals to operator by for satisfying condition and display results in form of subset.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Take a new variable as loc and create a subset of Employee\_Information whose Location==”Pune”.
3. Print loc variable to get desire a result
4. Run all commands by pressing ctrl+enter.
5. Save the script using save option from file.

**Code and Result:-**

**Take a new variable as loc and create a subset of Employee\_Information whose Location==Pune.**

#Take a new variable as loc and create a subset of Employee\_Information whose Location==”Pune”.

loc<-subset(Employee\_Information,Location==”Pune”)

**Print loc variable to get desire result**

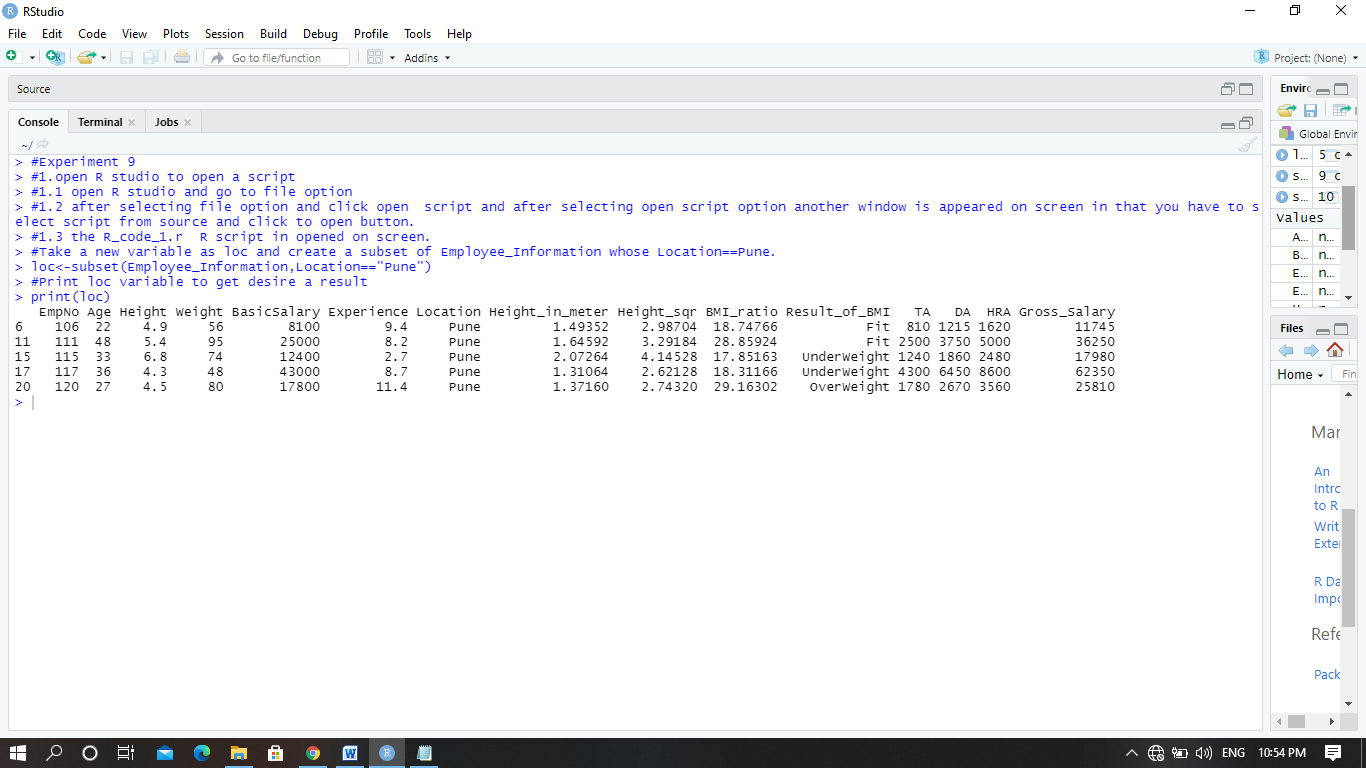
#Print loc variable to get desire result

print(loc)

**Save the script Excersice\_1.r on using save option.**

Save script with name Excersice\_1.r

**Output –**



**Conclusion –**

We can extract the values of element by satisfying conditions with help of relational operators exact equals to ‘== ’ and store their result in newly created subset.In case of checking condition using operator by passing condition as a string value then the string value must be written in double quotes “”.

**10. Display employee details whose location other than Nashik.**

**How to extract data element value from data frame as per the condition using relational operator not equal to operator ‘!=’ and how to store a result in newly created subset?**

1. **Relational!= operator :**- A ‘==’ not equals to operator is used when display result when resulted values are not equals to expected values.
2. We can use subset () function to perform this kind of basic operations to get result in another subset of data.
3. For run all commands of script, select commands and press **ctrl+enter.**

**Introduction:**

The purpose of this experiment is to understand the concept to use of relational operator not equals to operator by for satisfying condition and display results in form of subset.

**Procedure:-**

1. Open R studio open Excersice\_1.r script.
2. Take a new variable as other and create a subset of Employee\_Information whose Location!=”Nashik”.
3. Print other variable to get desire a result
4. Run all commands by pressing ctrl+enter.
5. Save the script using save option from file.

**Code and Result:-**

**Open R studio and open Excersice\_1.r script**

#1.open R studio to open a script

#1.1 open R studio and go to file option

#1.2 after selecting file option and click open script and after selecting open script option another window is appeared on screen in that you have to select script from source and click to open button.

#1.3 the Excersice\_1.r R script in opened on screen.

**Take a new variable as other and create a subset of Employee\_Information whose Location!=”Nashik”.**

#Take a new variable as other and create a subset of Employee\_Information whose Location!=”Nashik”.

other<-subset(Employee\_Information,Location!=”Nashik”)

**Print other variable to get desire result**

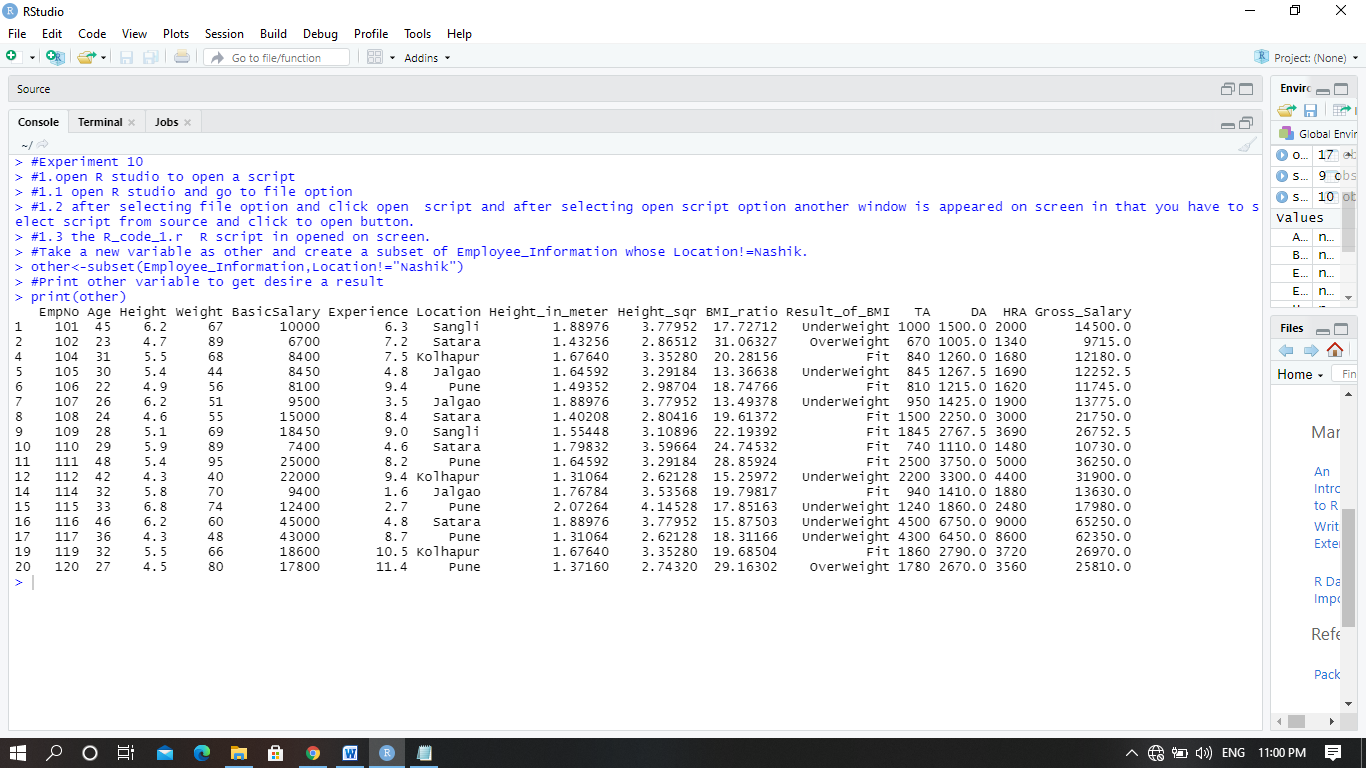
#Print other variable to get desire result

print(other)

**Save the script Excersice\_1.r on using save option..**

Save script with name Excersice\_1.r

**Output –**



**Conclusion –**

We can use the relational operator “!=” not equals to for excluding values in given condition. In case of checking condition using!= operator and the condition contains the string value then the string value must be written in double quotes .

**xxxxxxxxxxxxxx**