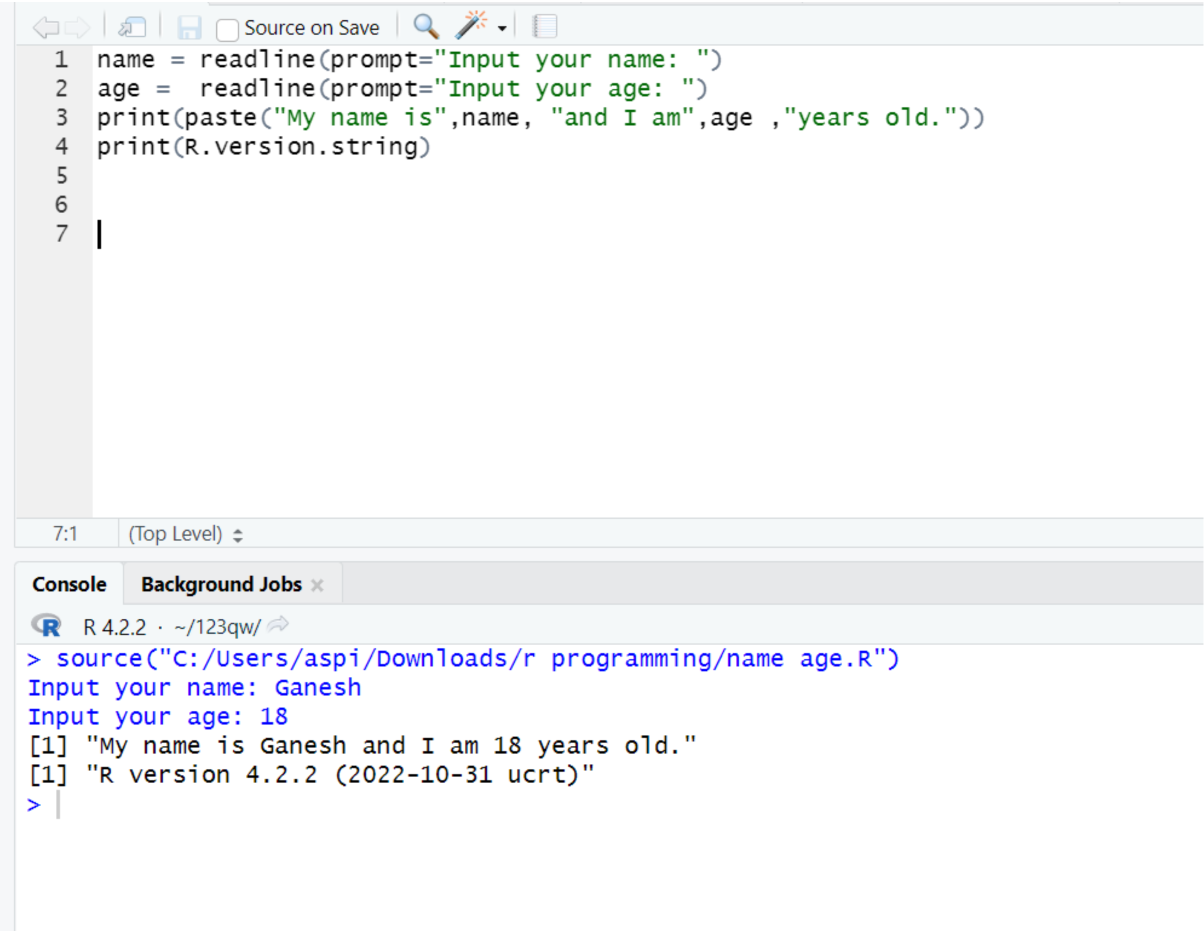
Basic R Prog Questions-Lab Day1

1. Write a R program to take input from the user (name and age) and display the values. Also print the version of R installation.

```
name = readline(prompt="Input your name: ")
age =  readline(prompt="Input your age: ")
print(paste("My name is",name, "and I am",age ,"years old."))
print(R.version.string)
```



```
name = readline(prompt="Input your name: ")
age =  readline(prompt="Input your age: ")
print(paste("My name is",name, "and I am",age ,"years old."))
print(R.version.string)
```

```
> source("C:/Users/aspi/Downloads/r programming/name age.R")
Input your name: Ganesh
Input your age: 18
[1] "My name is Ganesh and I am 18 years old."
[1] "R version 4.2.2 (2022-10-31 ucrt)"
>
```

2. Write a R program to get the details of the objects in memory.

name = "Python";

n1 = 10;

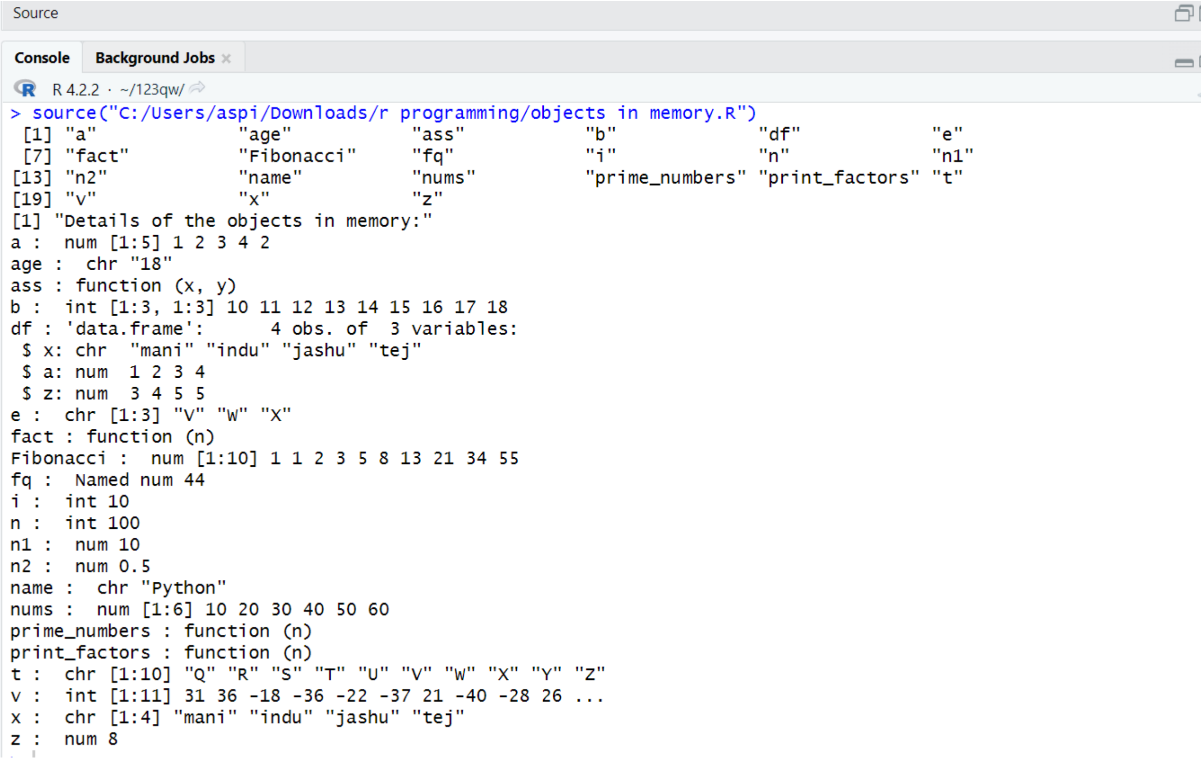
n2 = 0.5

nums = c(10, 20, 30, 40, 50, 60)

print(ls())

print("Details of the objects in memory:")

print(ls.str())



```
> source("C:/Users/aspi/Downloads/r programming/objects in memory.R")
 [1] "a"             "age"           "ass"           "b"             "df"            "e"            
 [7] "fact"          "Fibonacci"     "fq"            "i"             "n"             "n1"           
[13] "n2"            "name"          "nums"          "prime_numbers" "print_factors" "t"            
[19] "v"             "x"             "z"            
[1] "Details of the objects in memory:"
a :  num [1:5] 1 2 3 4 2
age :  chr "18"
ass : function (x, y)  
b :  int [1:3, 1:3] 10 11 12 13 14 15 16 17 18
df : 'data.frame':	4 obs. of  3 variables:
 $ x: chr  "mani" "indu" "jashu" "tej"
 $ a: num  1 2 3 4
 $ z: num  3 4 5 5
e :  chr [1:3] "V" "W" "X"
fact : function (n)  
Fibonacci :  num [1:10] 1 1 2 3 5 8 13 21 34 55
fq :  Named num 44
i :  int 10
n :  int 100
n1 :  num 10
n2 :  num 0.5
name :  chr "Python"
nums :  num [1:6] 10 20 30 40 50 60
prime_numbers : function (n)  
print_factors : function (n)  
t :  chr [1:10] "Q" "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
v :  int [1:11] 31 36 -18 -36 -22 -37 21 -40 -28 26 ...
x :  chr [1:4] "mani" "indu" "jashu" "tej"
z :  num 8
```

3. Write a R program to create a sequence of numbers from 20 to 50 and find the mean of numbers from 20 to 60 and sum of numbers from 51 to 91.

print("Sequence of numbers from 20 to 50:")

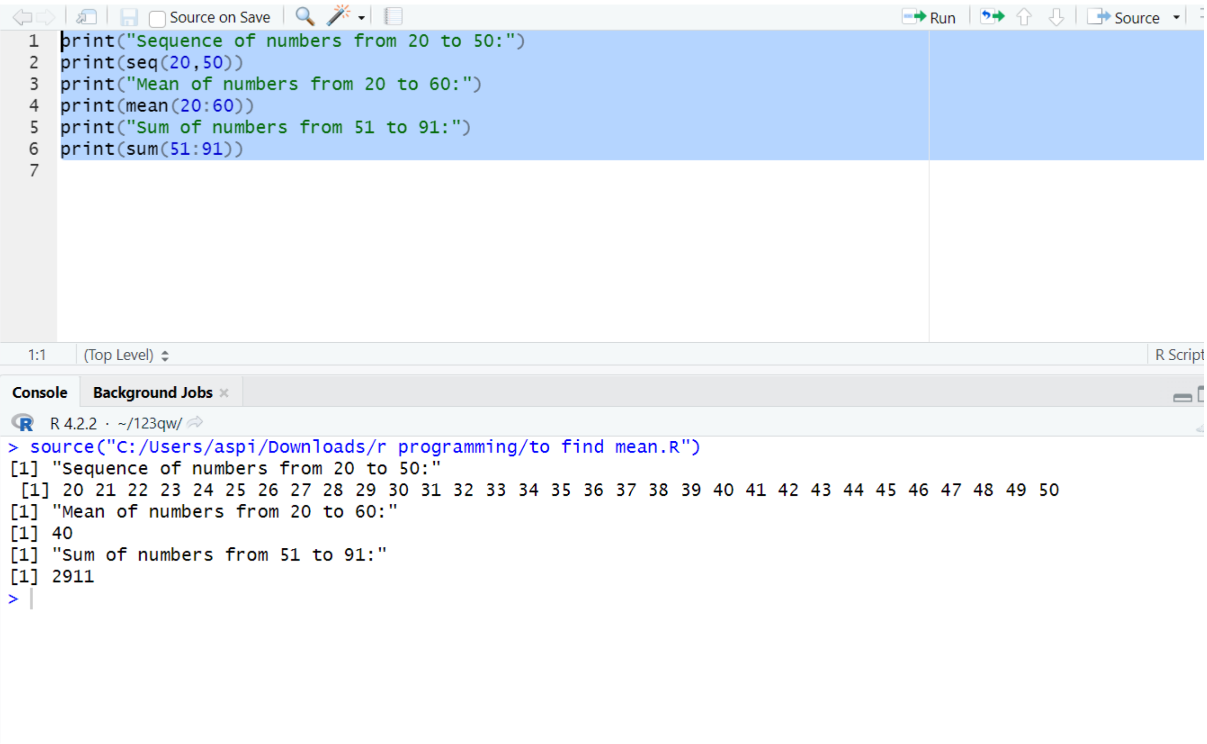
print(seq(20,50))

print("Mean of numbers from 20 to 60:")

print(mean(20:60))

print("Sum of numbers from 51 to 91:")

print(sum(51:91))

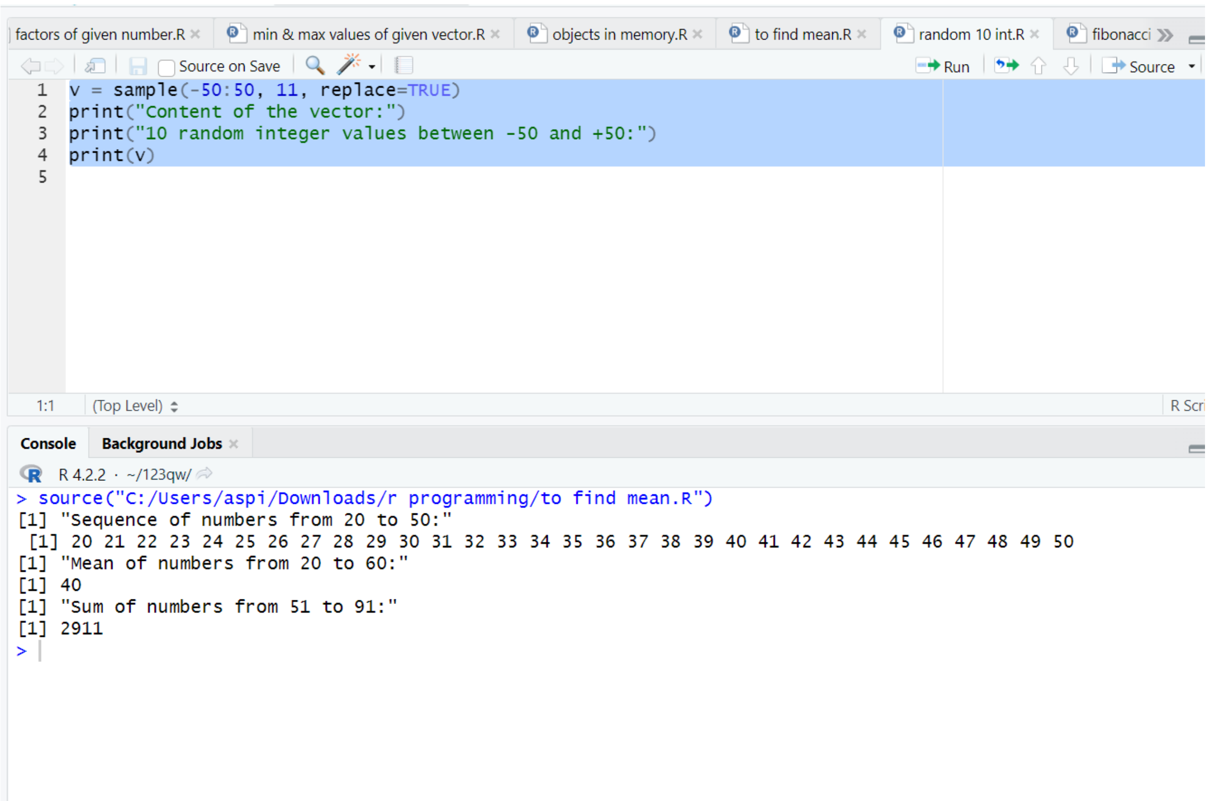


```
print("Sequence of numbers from 20 to 50:")
print(seq(20,50))
print("Mean of numbers from 20 to 60:")
print(mean(20:60))
print("Sum of numbers from 51 to 91:")
print(sum(51:91))
```

```
> source("C:/Users/aspi/Downloads/r programming/to find mean.R")
[1] "Sequence of numbers from 20 to 50:"
 [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
[1] "Mean of numbers from 20 to 60:"
[1] 40
[1] "Sum of numbers from 51 to 91:"
[1] 2911
> 
```

4. Write a R program to create a vector which contains 10 random integer values between -50 and +50.

```
v = sample(-50:50, 11, replace=TRUE)

print("Content of the vector:")

print("10 random integer values between -50 and +50:")

print(v)
```

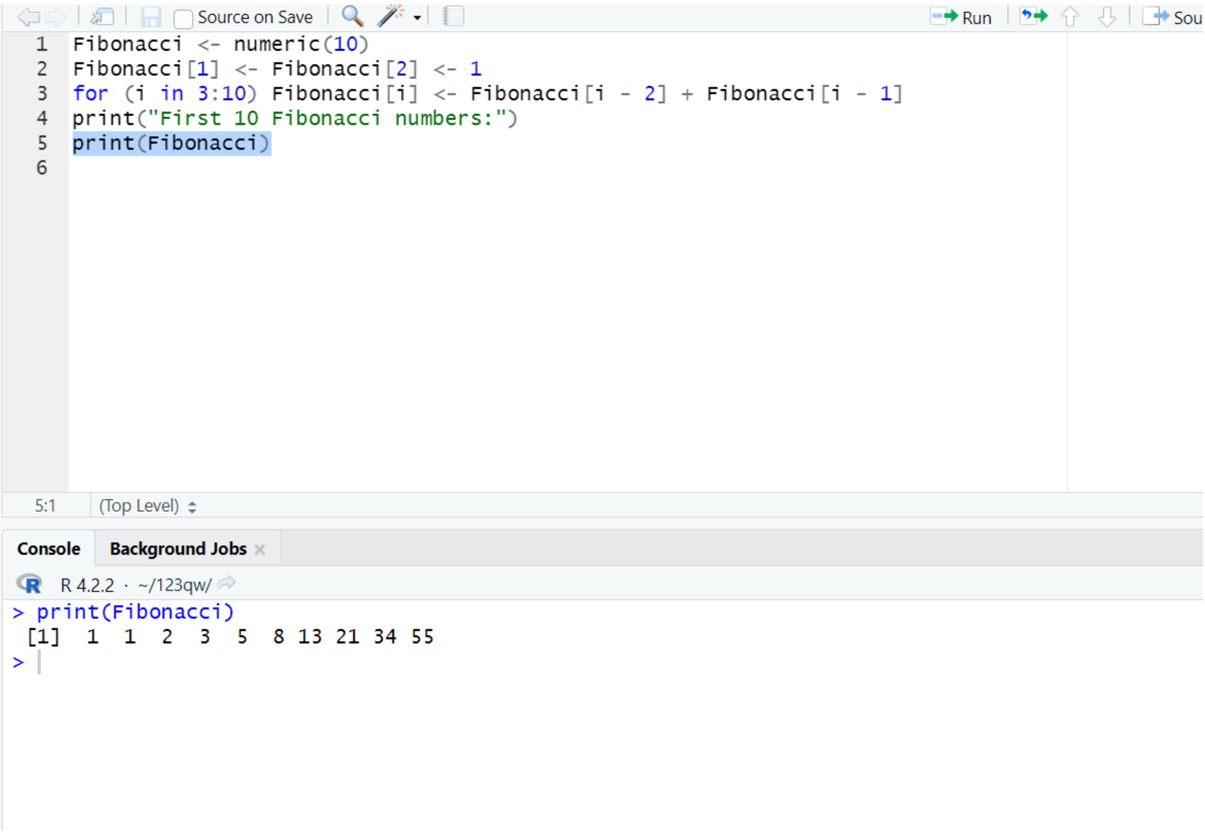


```r
v = sample(-50:50, 11, replace=TRUE)
print("Content of the vector:")
print("10 random integer values between -50 and +50:")
print(v)
```

```
> source("C:/Users/aspi/Downloads/r programming/to find mean.R")
[1] "Sequence of numbers from 20 to 50:"
 [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
[1] "Mean of numbers from 20 to 60:"
[1] 40
[1] "Sum of numbers from 51 to 91:"
[1] 2911
> 
```

5. Write a R program to get the first 10 Fibonacci numbers.

```r
Fibonacci <- numeric(10)

Fibonacci[1] <- Fibonacci[2] <- 1

for (i in 3:10) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]

print("First 10 Fibonacci numbers:")

print(Fibonacci)
```

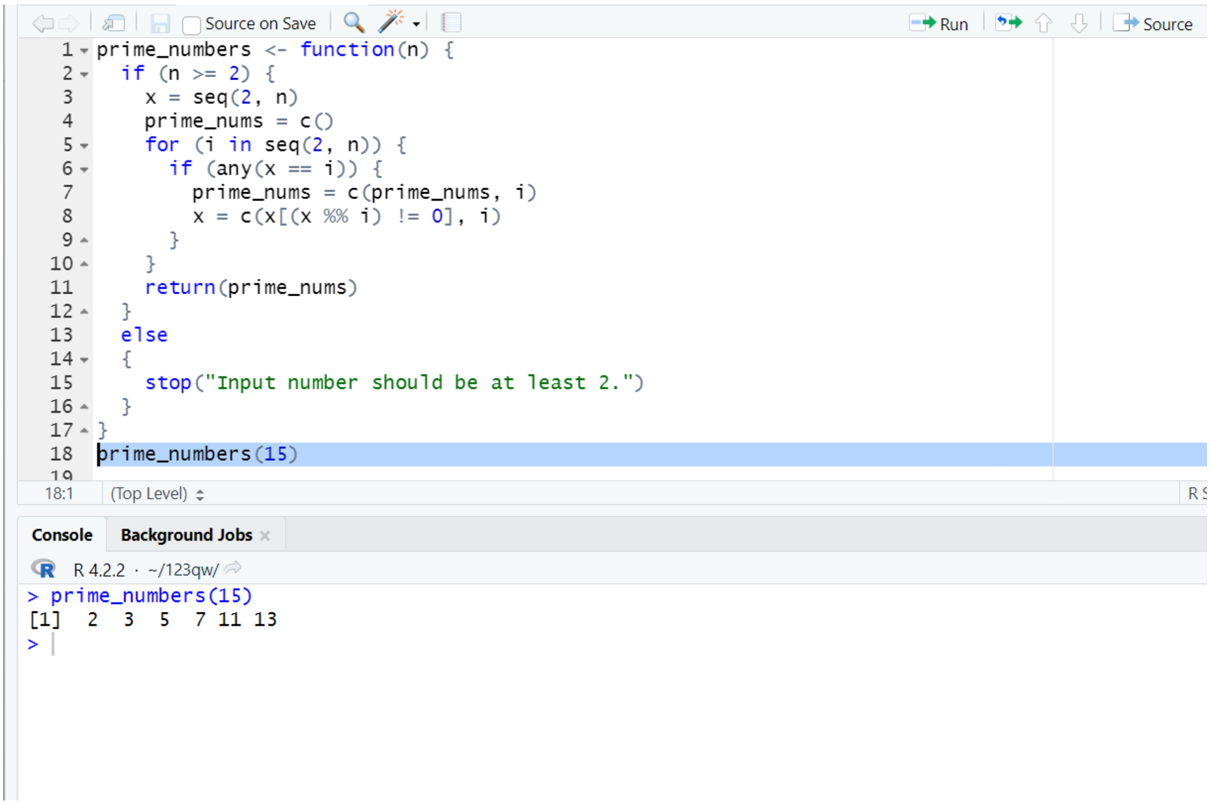


```
Fibonacci <- numeric(10)
Fibonacci[1] <- Fibonacci[2] <- 1
for (i in 3:10) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]
print("First 10 Fibonacci numbers:")
print(Fibonacci)
```

```
> print(Fibonacci)
 [1]  1  1  2  3  5  8 13 21 34 55
> 
```

6. Write a R program to get all prime numbers up to a given number (based on the sieve of Eratosthenes).

```
prime_numbers <- function(n) {
 if (n >= 2) {
  x = seq(2, n)
  prime_nums = c()
  for (i in seq(2, n)) {
   if (any(x == i)) {
     prime_nums = c(prime_nums, i)
     x = c(x[(x %% i) != 0], i)
   }
  }
  return(prime_nums)
 }
 else
 {
  stop("Input number should be at least 2.")
 }
}
prime_numbers(15)
```



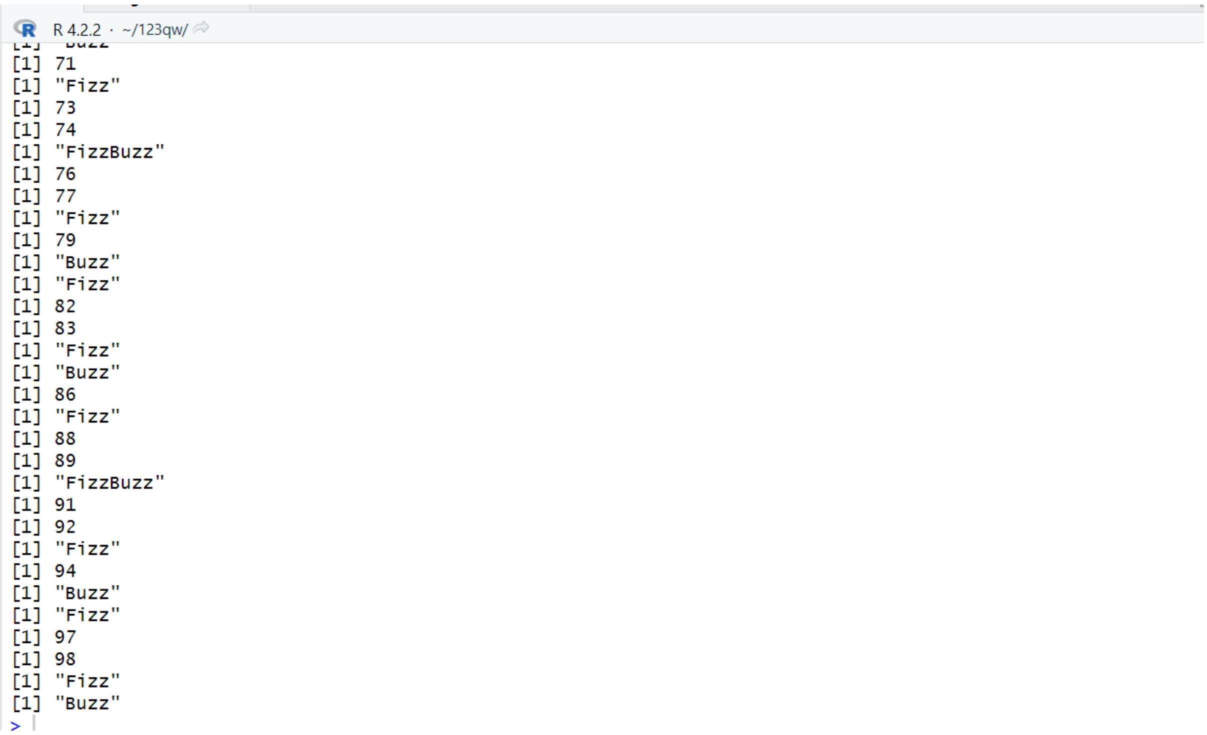
```
prime_numbers <- function(n) {
  if (n >= 2) {
    x = seq(2, n)
    prime_nums = c()
    for (i in seq(2, n)) {
      if (any(x == i)) {
        prime_nums = c(prime_nums, i)
        x = c(x[(x %% i) != 0], i)
      }
    }
    return(prime_nums)
  }
  else
  {
    stop("Input number should be at least 2.")
  }
}
prime_numbers(15)
```

```
> prime_numbers(15)
[1]  2  3  5  7 11 13
> 
```

7. Write a R program to print the numbers from 1 to 100 and print &quot;Fizz&quot; for multiples of 3, print &quot;Buzz&quot; for multiples of 5, and print &quot;FizzBuzz&quot; for multiples of

both.

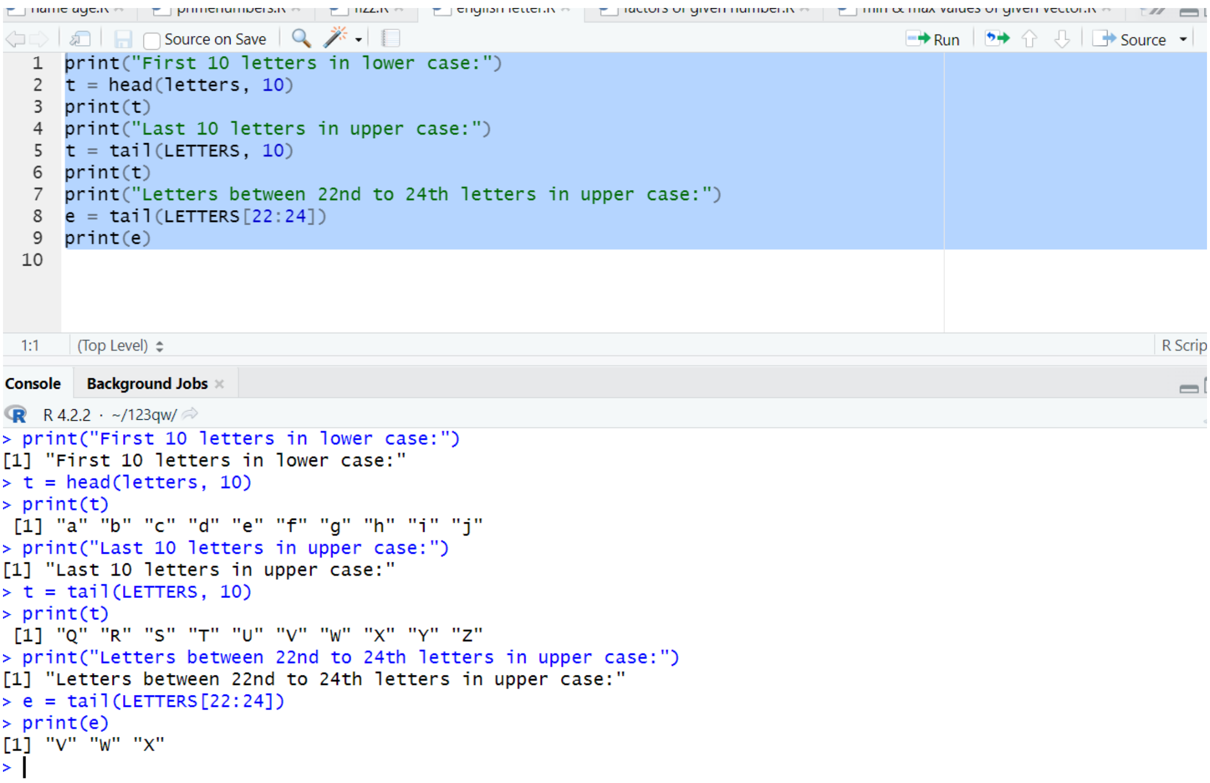
```
for (n in 1:100) {
  if (n %% 3 == 0 & n %% 5 == 0) {print("FizzBuzz")}
  else if (n %% 3 == 0) {print("Fizz")}
  else if (n %% 5 == 0) {print("Buzz")}
  else print(n)
}
```



```
[1] "Buzz"
[1] 71
[1] "Fizz"
[1] 73
[1] 74
[1] "FizzBuzz"
[1] 76
[1] 77
[1] "Fizz"
[1] 79
[1] "Buzz"
[1] "Fizz"
[1] 82
[1] 83
[1] "Fizz"
[1] "Buzz"
[1] 86
[1] "Fizz"
[1] 88
[1] 89
[1] "FizzBuzz"
[1] 91
[1] 92
[1] "Fizz"
[1] 94
[1] "Buzz"
[1] "Fizz"
[1] 97
[1] 98
[1] "Fizz"
[1] "Buzz"
>
```

8. Write a R program to extract first 10 english letter in lower case and last 10 letters in upper case and extract letters between 22 nd to 24 th letters in upper case.

```
print("First 10 letters in lower case:")

t = head(letters, 10)

print(t)

print("Last 10 letters in upper case:")

t = tail(LETTERS, 10)

print(t)

print("Letters between 22nd to 24th letters in upper case:")

e = tail(LETTERS[22:24])

print(e)
```

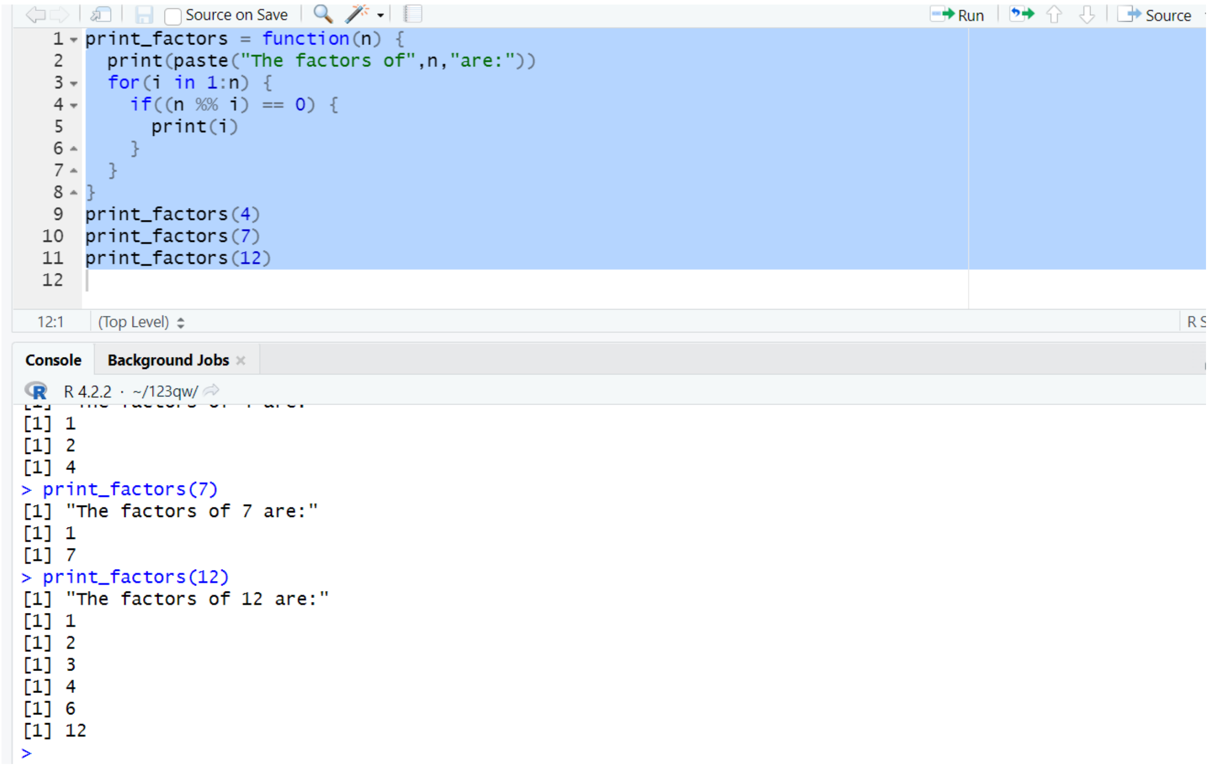


```r
print("First 10 letters in lower case:")
t = head(letters, 10)
print(t)
print("Last 10 letters in upper case:")
t = tail(LETTERS, 10)
print(t)
print("Letters between 22nd to 24th letters in upper case:")
e = tail(LETTERS[22:24])
print(e)
```

```
> print("First 10 letters in lower case:")
[1] "First 10 letters in lower case:"
> t = head(letters, 10)
> print(t)
 [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
> print("Last 10 letters in upper case:")
[1] "Last 10 letters in upper case:"
> t = tail(LETTERS, 10)
> print(t)
 [1] "Q" "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
> print("Letters between 22nd to 24th letters in upper case:")
[1] "Letters between 22nd to 24th letters in upper case:"
> e = tail(LETTERS[22:24])
> print(e)
[1] "V" "W" "X"
> 
```

9. Write a R program to find the factors of a given number.

```
print_factors = function(n) {
  print(paste("The factors of",n,"are:"))
  for(i in 1:n) {
    if((n %% i) == 0) {
      print(i)
    }
  }
}
print_factors(4)
print_factors(7)
print_factors(12)
```



```
print_factors = function(n) {
  print(paste("The factors of",n,"are:"))
  for(i in 1:n) {
    if((n %% i) == 0) {
      print(i)
    }
  }
}
print_factors(4)
print_factors(7)
print_factors(12)
```

```
[1] "The factors of 4 are:"
[1] 1
[1] 2
[1] 4
> print_factors(7)
[1] "The factors of 7 are:"
[1] 1
[1] 7
> print_factors(12)
[1] "The factors of 12 are:"
[1] 1
[1] 2
[1] 3
[1] 4
[1] 6
[1] 12
>
```

10. Write a R program to find the maximum and the minimum value of a given vector.

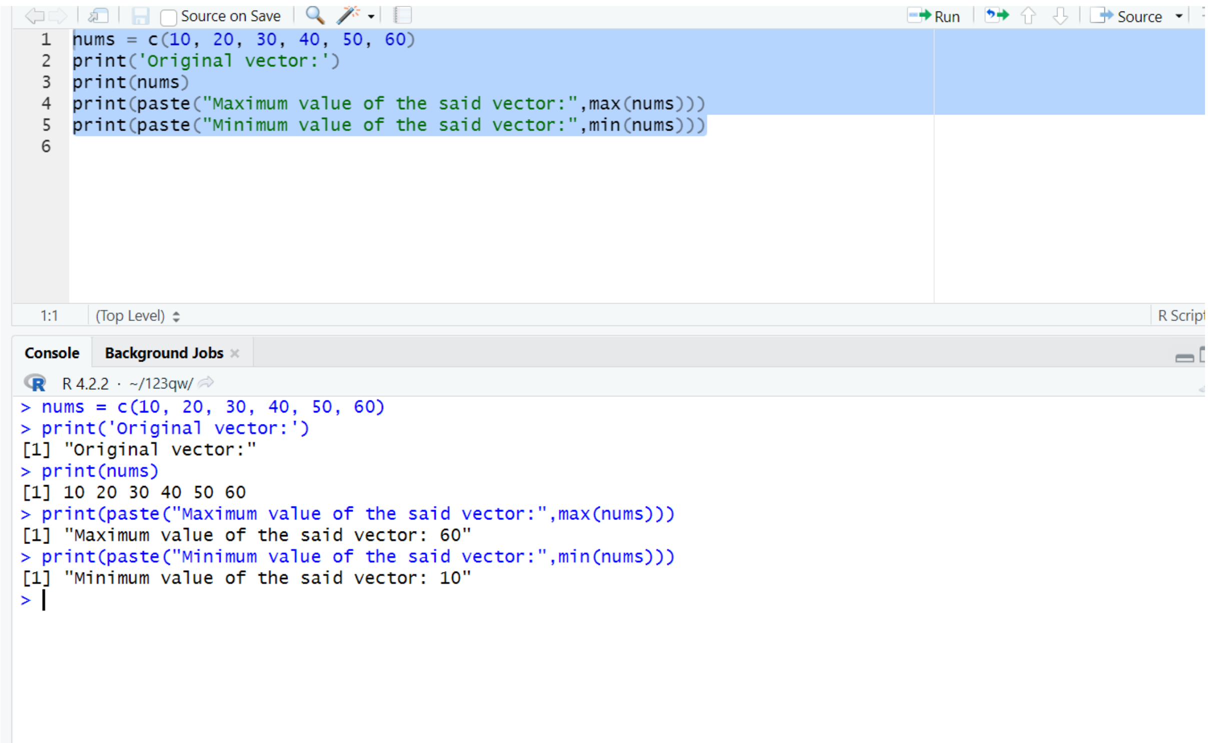
nums = c(10, 20, 30, 40, 50, 60)

print('Original vector:')

print(nums)

print(paste("Maximum value of the said vector:",max(nums)))

print(paste("Minimum value of the said vector:",min(nums)))



```
nums = c(10, 20, 30, 40, 50, 60)
print('Original vector:')
print(nums)
print(paste("Maximum value of the said vector:",max(nums)))
print(paste("Minimum value of the said vector:",min(nums)))
```

```
> nums = c(10, 20, 30, 40, 50, 60)
> print('Original vector:')
[1] "Original vector:"
> print(nums)
[1] 10 20 30 40 50 60
> print(paste("Maximum value of the said vector:",max(nums)))
[1] "Maximum value of the said vector: 60"
> print(paste("Minimum value of the said vector:",min(nums)))
[1] "Minimum value of the said vector: 10"
> 
```

11. Write a R program to get the unique elements of a given string and unique numbers of vector.

str1 = "The quick brown fox jumps over the lazy dog."

print("Original vector(string)")

print(str1)

print("Unique elements of the said vector:")

print(unique(tolower(str1)))

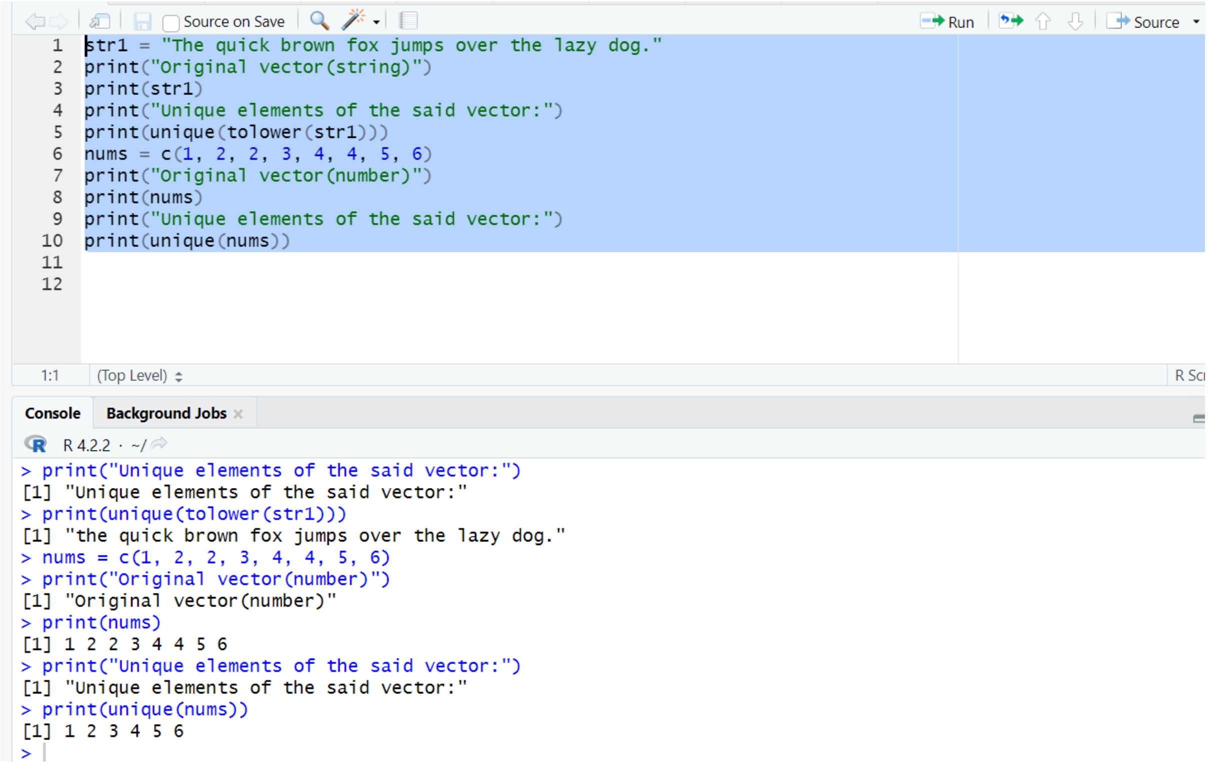
nums = c(1, 2, 2, 3, 4, 4, 5, 6)

print("Original vector(number)")

print(nums)

print("Unique elements of the said vector:")

print(unique(nums))

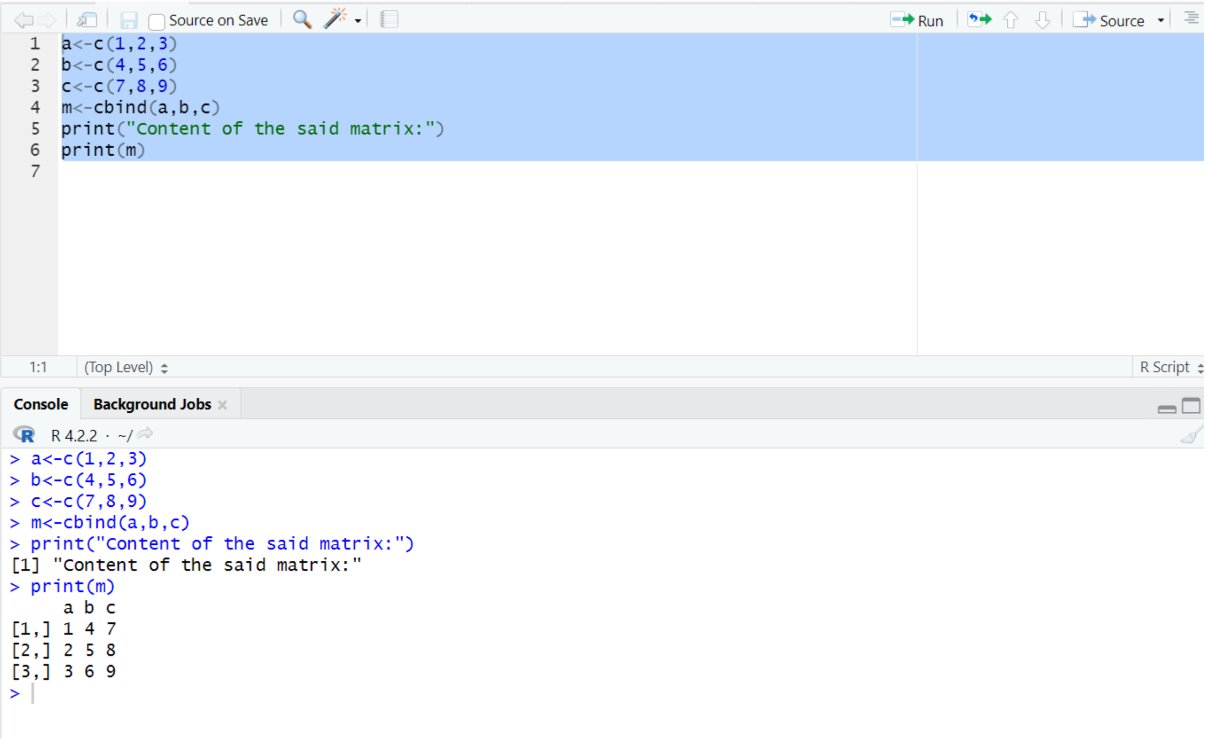


```
str1 = "The quick brown fox jumps over the lazy dog."
print("Original vector(string)")
print(str1)
print("Unique elements of the said vector:")
print(unique(tolower(str1)))
nums = c(1, 2, 2, 3, 4, 4, 5, 6)
print("Original vector(number)")
print(nums)
print("Unique elements of the said vector:")
print(unique(nums))
```

```
> print("Unique elements of the said vector:")
[1] "Unique elements of the said vector:"
> print(unique(tolower(str1)))
[1] "the quick brown fox jumps over the lazy dog."
> nums = c(1, 2, 2, 3, 4, 4, 5, 6)
> print("Original vector(number)")
[1] "Original vector(number)"
> print(nums)
[1] 1 2 2 3 4 4 5 6
> print("Unique elements of the said vector:")
[1] "Unique elements of the said vector:"
> print(unique(nums))
[1] 1 2 3 4 5 6
> 
```

12. Write a R program to create three vectors a,b,c with 3 integers. Combine the three vectors to become a 3×3 matrix where each column represents a vector. Print the content of the matrix.

```
a<-c(1,2,3)

b<-c(4,5,6)

c<-c(7,8,9)

m<-cbind(a,b,c)

print("Content of the said matrix:")

print(m)
```

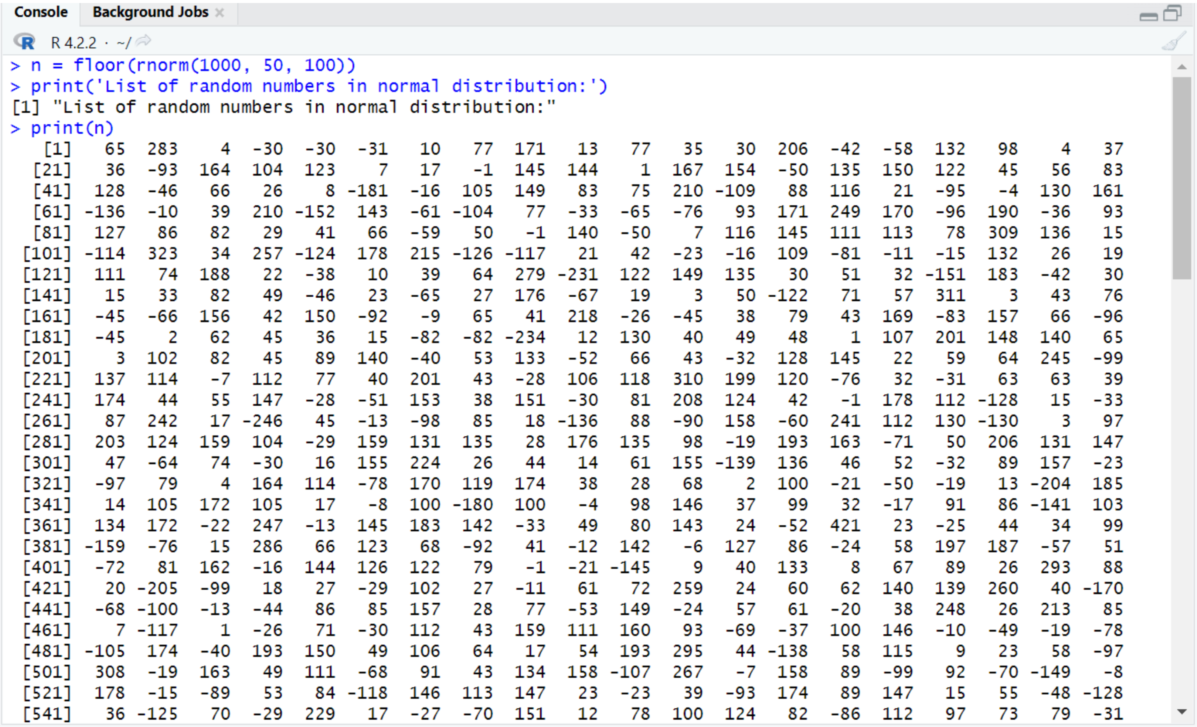


```
a<-c(1,2,3)
b<-c(4,5,6)
c<-c(7,8,9)
m<-cbind(a,b,c)
print("Content of the said matrix:")
print(m)
```

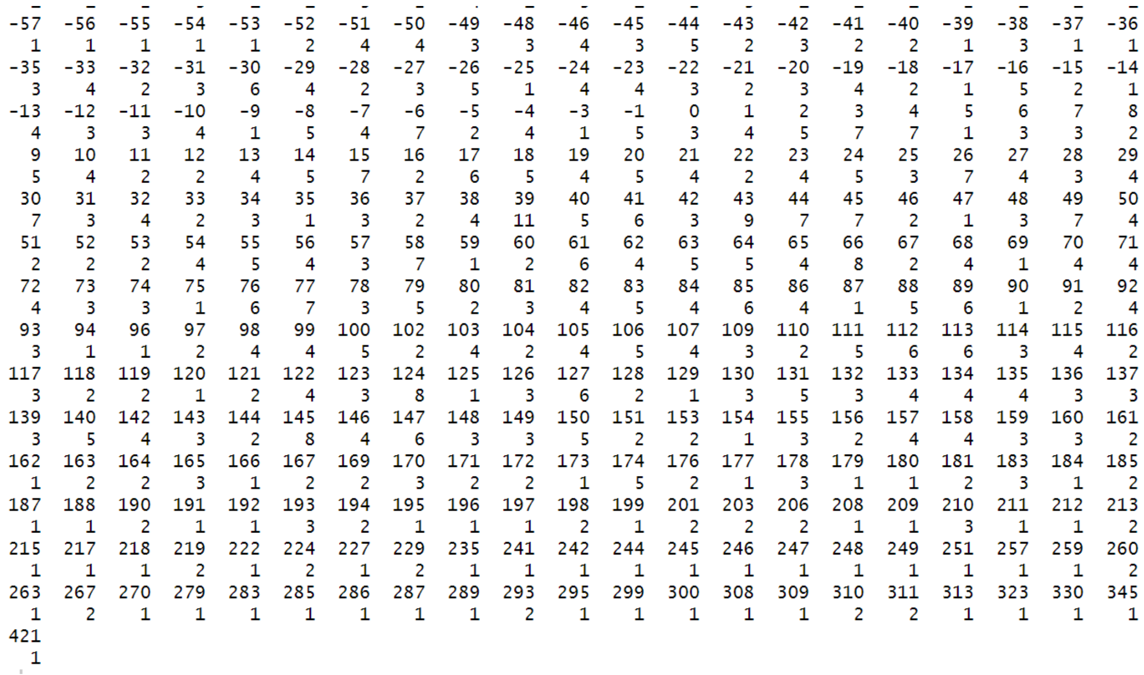
```
> a<-c(1,2,3)
> b<-c(4,5,6)
> c<-c(7,8,9)
> m<-cbind(a,b,c)
> print("Content of the said matrix:")
[1] "Content of the said matrix:"
> print(m)
     a b c
[1,] 1 4 7
[2,] 2 5 8
[3,] 3 6 9
> 
```

13. Write a R program to create a list of random numbers in normal distribution and count occurrences of each value.

```
n = floor(rnorm(1000, 50, 100))

print('List of random numbers in normal distribution:')

print(n)

t = table(n)

print("Count occurrences of each value:")

print(t)
```



```
> n = floor(rnorm(1000, 50, 100))
> print('List of random numbers in normal distribution:')
[1] "List of random numbers in normal distribution:"
> print(n)
   [1]   65  283    4  -30  -30  -31   10   77  171   13   77   35   30  206  -42  -58  132   98    4   37
  [21]   36  -93  164  104  123    7   17   -1  145  144    1  167  154  -50  135  150  122   45   56   83
  [41]  128  -46   66   26    8 -181  -16  105  149   83   75  210 -109   88  116   21  -95   -4  130  161
  [61] -136  -10   39  210 -152  143  -61 -104   77  -33  -65  -76   93  171  249  170  -96  190  -36   93
  [81]  127   86   82   29   41   66  -59   50   -1  140  -50    7  116  145  111  113   78  309  136   15
 [101] -114  323   34  257 -124  178  215 -126 -117   21   42  -23  -16  109  -81  -11  -15  132   26   19
 [121]  111   74  188   22  -38   10   39   64  279 -231  122  149  135   30   51   32 -151  183  -42   30
 [141]   15   33   82   49  -46   23  -65   27  176  -67   19    3   50 -122   71   57  311    3   43   76
 [161]  -45  -66  156   42  150  -92   -9   65   41  218  -26  -45   38   79   43  169  -83  157   66  -96
 [181]  -45    2   62   45   36   15  -82  -82 -234   12  130   40   49   48    1  107  201  148  140   65
 [201]    3  102   82   45   89  140  -40   53  133  -52   66   43  -32  128  145   22   59   64  245  -99
 [221]  137  114   -7  112   77   40  201   43  -28  106  118  310  199  120  -76   32  -31   63   63   39
 [241]  174   44   55  147  -28  -51  153   38  151  -30   81  208  124   42   -1  178  112 -128   15  -33
 [261]   87  242   17 -246   45  -13  -98   85   18 -136   88  -90  158  -60  241  112  130 -130    3   97
 [281]  203  124  159  104  -29  159  131  135   28  176  135   98  -19  193  163  -71   50  206  131  147
 [301]   47  -64   74  -30   16  155  224   26   44   14   61  155 -139  136   46   52  -32   89  157  -23
 [321]  -97   79    4  164  114  -78  170  119  174   38   28   68    2  100  -21  -50  -19   13 -204  185
 [341]   14  105  172  105   17   -8  100 -180  100   -4   98  146   37   99   32  -17   91   86 -141  103
 [361]  134  172  -22  247  -13  145  183  142  -33   49   80  143   24  -52  421   23  -25   44   34   99
 [381] -159  -76   15  286   66  123   68  -92   41  -12  142   -6  127   86  -24   58  197  187  -57   51
 [401]  -72   81  162  -16  144  126  122   79   -1  -21 -145    9   40  133    8   67   89   26  293   88
 [421]   20 -205  -99   18   27  -29  102   27  -11   61   72  259   24   60   62  140  139  260   40 -170
 [441]  -68 -100  -13  -44   86   85  157   28   77  -53  149  -24   57   61  -20   38  248   26  213   85
 [461]    7 -117    1  -26   71  -30  112   43  159  111  160   93  -69  -37  100  146  -10  -49  -19  -78
 [481] -105  174  -40  193  150   49  106   64   17   54  193  295   44 -138   58  115    9   23   58  -97
 [501]  308  -19  163   49  111  -68   91   43  134  158 -107  267   -7  158   89  -99   92  -70 -149   -8
 [521]  178  -15  -89   53   84 -118  146  113  147   23  -23   39  -93  174   89  147   15   55  -48 -128
 [541]   36 -125   70  -29  229   17  -27  -70  151   12   78  100  124   82  -86  112   97   73   79  -31
```



```
 -57  -56  -55  -54  -53  -52  -51  -50  -49  -48  -46  -45  -44  -43  -42  -41  -40  -39  -38  -37  -36 
   1    1    1    1    1    2    4    4    3    3    4    3    5    2    3    2    2    1    3    1    1 
 -35  -33  -32  -31  -30  -29  -28  -27  -26  -25  -24  -23  -22  -21  -20  -19  -18  -17  -16  -15  -14 
   3    4    2    3    6    4    2    3    5    1    4    4    3    2    3    4    2    1    5    2    1 
 -13  -12  -11  -10   -9   -8   -7   -6   -5   -4   -3   -1    0    1    2    3    4    5    6    7    8 
   4    3    3    4    1    5    4    7    2    4    1    5    3    4    5    7    7    1    3    3    2 
   9   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24   25   26   27   28   29 
   5    4    2    2    4    5    7    2    6    5    4    5    4    2    4    5    3    7    4    3    4 
  30   31   32   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48   49   50 
   7    3    4    2    3    1    3    2    4   11    5    6    3    9    7    7    2    1    3    7    4 
  51   52   53   54   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69   70   71 
   2    2    2    4    5    4    3    7    1    2    6    4    5    5    4    8    2    4    1    4    4 
  72   73   74   75   76   77   78   79   80   81   82   83   84   85   86   87   88   89   90   91   92 
   4    3    3    1    6    7    3    5    2    3    4    5    4    6    4    1    5    6    1    2    4 
  93   94   96   97   98   99  100  102  103  104  105  106  107  109  110  111  112  113  114  115  116 
   3    1    1    2    4    4    5    2    4    2    4    5    4    3    2    5    6    6    3    4    2 
 117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137 
   3    2    2    1    2    4    3    8    1    3    6    2    1    3    5    3    4    4    4    3    3 
 139  140  142  143  144  145  146  147  148  149  150  151  153  154  155  156  157  158  159  160  161 
   3    5    4    3    2    8    4    6    3    3    5    2    2    1    3    2    4    4    3    3    2 
 162  163  164  165  166  167  169  170  171  172  173  174  176  177  178  179  180  181  183  184  185 
   1    2    2    3    1    2    2    3    2    2    1    5    2    1    3    1    1    2    3    1    2 
 187  188  190  191  192  193  194  195  196  197  198  199  201  203  206  208  209  210  211  212  213 
   1    1    2    1    1    3    2    1    1    1    2    1    2    2    2    1    1    3    1    1    2 
 215  217  218  219  222  224  227  229  235  241  242  244  245  246  247  248  249  251  257  259  260 
   1    1    1    2    1    2    1    2    1    1    1    1    1    1    1    1    1    1    1    1    2 
 263  267  270  279  283  285  286  287  289  293  295  299  300  308  309  310  311  313  323  330  345 
   1    2    1    1    1    1    1    1    1    2    1    1    1    1    1    2    2    1    1    1    1 
 421 
   1 
```

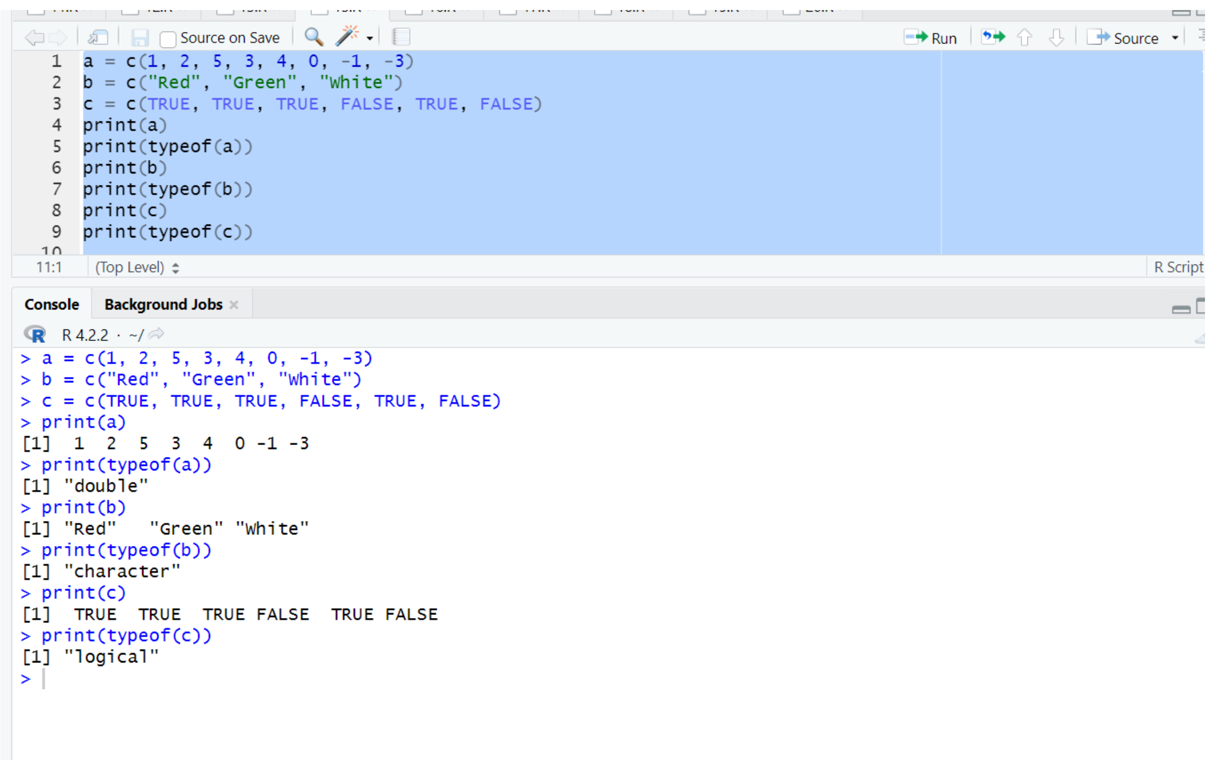
15. Write a R program to create three vectors numeric data, character data and logical data. Display the content of the vectors and their type.

a = c(1, 2, 5, 3, 4, 0, -1, -3)

b = c("Red", "Green", "White")

c = c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)

```
print(a)

print(typeof(a))

print(b)

print(typeof(b))

print(c)

print(typeof(c))
```



```
a = c(1, 2, 5, 3, 4, 0, -1, -3)
b = c("Red", "Green", "White")
c = c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)
print(a)
print(typeof(a))
print(b)
print(typeof(b))
print(c)
print(typeof(c))
```

```
> a = c(1, 2, 5, 3, 4, 0, -1, -3)
> b = c("Red", "Green", "White")
> c = c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)
> print(a)
[1]  1  2  5  3  4  0 -1 -3
> print(typeof(a))
[1] "double"
> print(b)
[1] "Red"   "Green" "White"
> print(typeof(b))
[1] "character"
> print(c)
[1]  TRUE  TRUE  TRUE FALSE  TRUE FALSE
> print(typeof(c))
[1] "logical"
> 
```

16. Write a R program to create a 5 x 4 matrix , 3 x 3 matrix with labels and fill the matrix by rows and 2 × 2 matrix with labels and fill the matrix by columns.

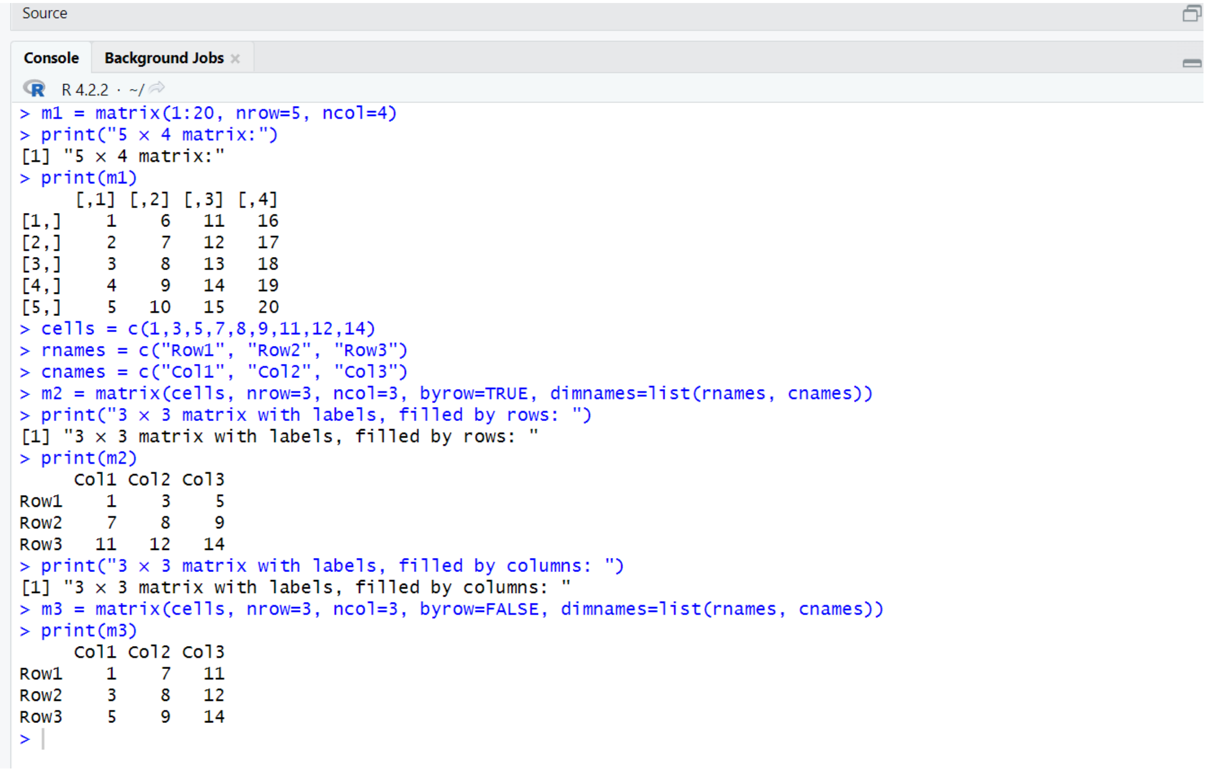
```
m1 = matrix(1:20, nrow=5, ncol=4)

print("5 × 4 matrix:")

print(m1)

cells = c(1,3,5,7,8,9,11,12,14)

rnames = c("Row1", "Row2", "Row3")

cnames = c("Col1", "Col2", "Col3")

m2 = matrix(cells, nrow=3, ncol=3, byrow=TRUE, dimnames=list(rnames, cnames))

print("3 × 3 matrix with labels, filled by rows: ")
```

print(m2)

print("3 × 3 matrix with labels, filled by columns: ")

m3 = matrix(cells, nrow=3, ncol=3, byrow=FALSE, dimnames=list(rnames, cnames))

print(m3)



```
> m1 = matrix(1:20, nrow=5, ncol=4)
> print("5 × 4 matrix:")
[1] "5 × 4 matrix:"
> print(m1)
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
> cells = c(1,3,5,7,8,9,11,12,14)
> rnames = c("Row1", "Row2", "Row3")
> cnames = c("Col1", "Col2", "Col3")
> m2 = matrix(cells, nrow=3, ncol=3, byrow=TRUE, dimnames=list(rnames, cnames))
> print("3 × 3 matrix with labels, filled by rows: ")
[1] "3 × 3 matrix with labels, filled by rows: "
> print(m2)
     Col1 Col2 Col3
Row1    1    3    5
Row2    7    8    9
Row3   11   12   14
> print("3 × 3 matrix with labels, filled by columns: ")
[1] "3 × 3 matrix with labels, filled by columns: "
> m3 = matrix(cells, nrow=3, ncol=3, byrow=FALSE, dimnames=list(rnames, cnames))
> print(m3)
     Col1 Col2 Col3
Row1    1    7   11
Row2    3    8   12
Row3    5    9   14
> 
```

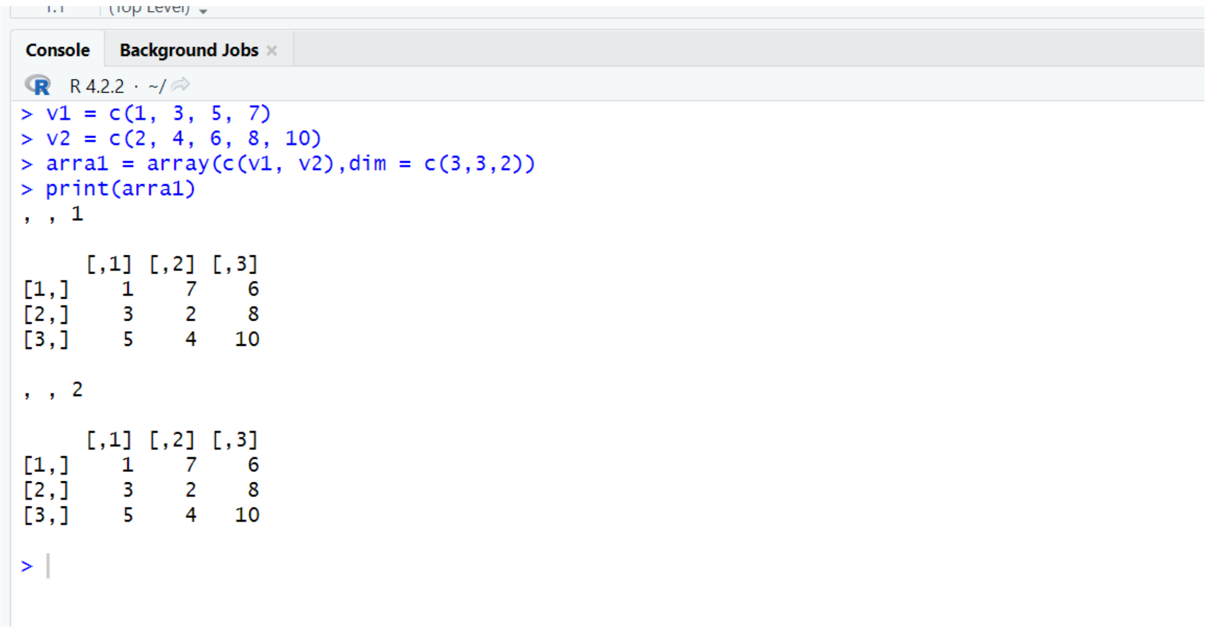
18. Write a R program to create an array with three columns, three rows, and two &quot;tables&quot;, taking two vectors as input to the array. Print the array.

v1 = c(1, 3, 5, 7)

v2 = c(2, 4, 6, 8, 10)

arra1 = array(c(v1, v2),dim = c(3,3,2))

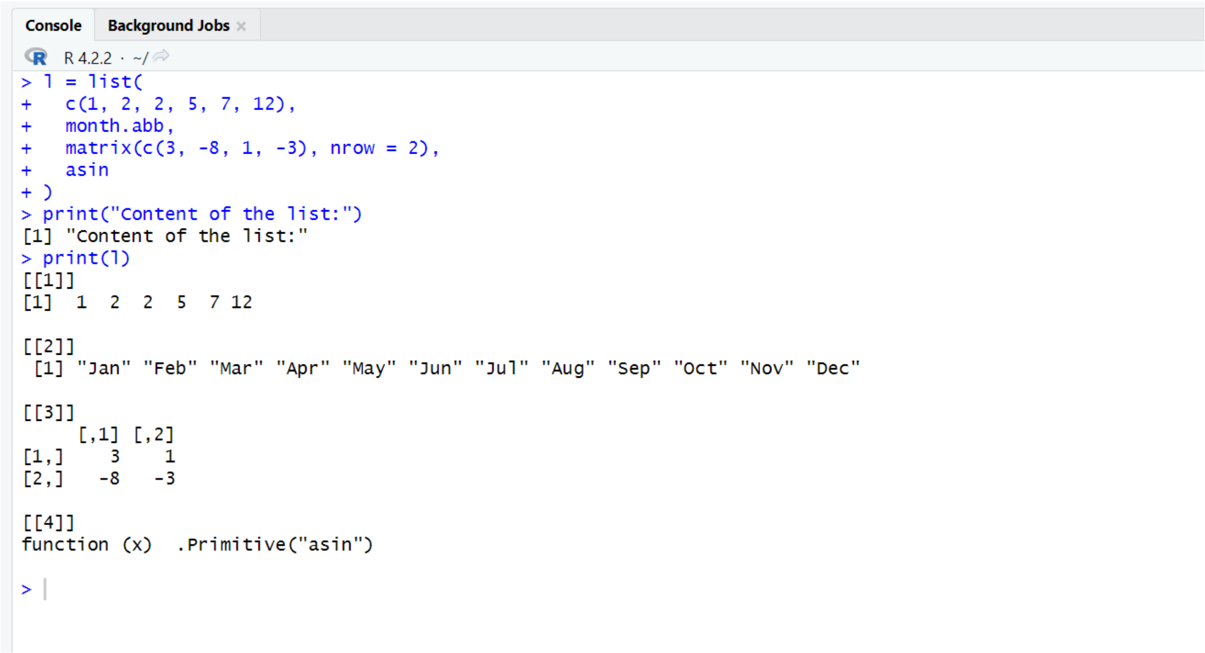
print(arra1)



```
> v1 = c(1, 3, 5, 7)
> v2 = c(2, 4, 6, 8, 10)
> arra1 = array(c(v1, v2),dim = c(3,3,2))
> print(arra1)
, , 1

     [,1] [,2] [,3]
[1,]    1    7    6
[2,]    3    2    8
[3,]    5    4   10

, , 2

     [,1] [,2] [,3]
[1,]    1    7    6
[2,]    3    2    8
[3,]    5    4   10

> 
```

19. Write a R program to create a list of elements using vectors, matrices and a functions. Print the content of the list.

```
l = list(
  c(1, 2, 2, 5, 7, 12),
  month.abb,
  matrix(c(3, -8, 1, -3), nrow = 2),
  asin
)
print("Content of the list:")
print(l)
```

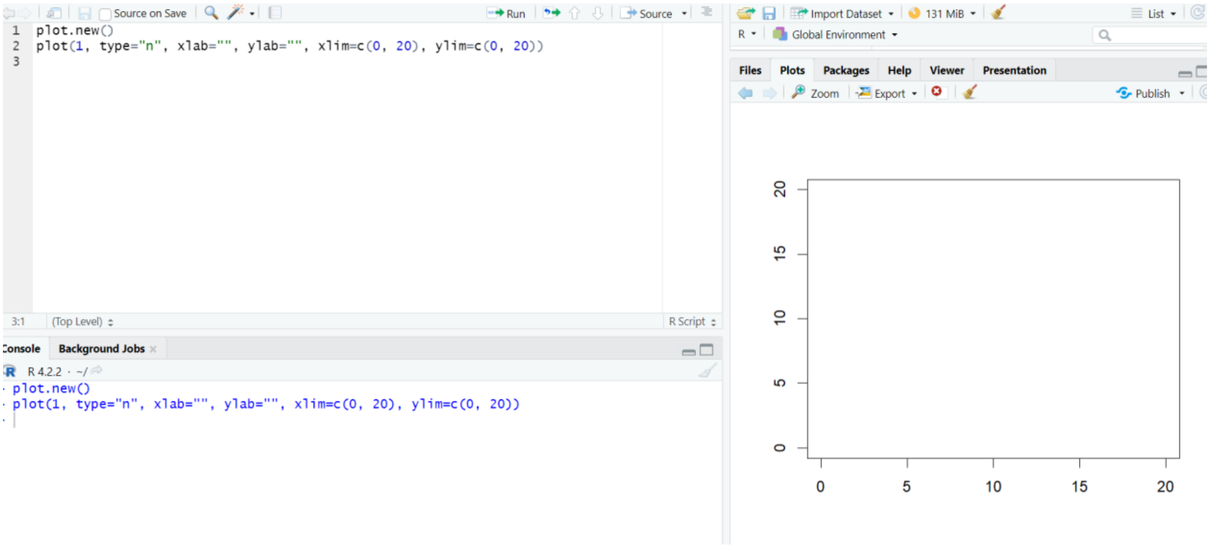


```
> l = list(
+   c(1, 2, 2, 5, 7, 12),
+   month.abb,
+   matrix(c(3, -8, 1, -3), nrow = 2),
+   asin
+ )
> print("Content of the list:")
[1] "Content of the list:"
> print(l)
[[1]]
[1]  1  2  2  5  7 12

[[2]]
 [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"

[[3]]
     [,1] [,2]
[1,]    3    1
[2,]   -8   -3

[[4]]
function (x)  .Primitive("asin")

> 
```

20. Write a R program to draw an empty plot and an empty plot specify the axes limits of the graphic

plot.new()

plot(1, type="n", xlab="", ylab="", xlim=c(0, 20), ylim=c(0, 20))



```
plot.new()
plot(1, type="n", xlab="", ylab="", xlim=c(0, 20), ylim=c(0, 20))
```

```
> plot.new()
> plot(1, type="n", xlab="", ylab="", xlim=c(0, 20), ylim=c(0, 20))
> 
```