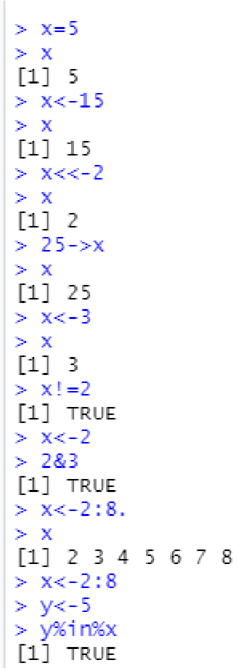
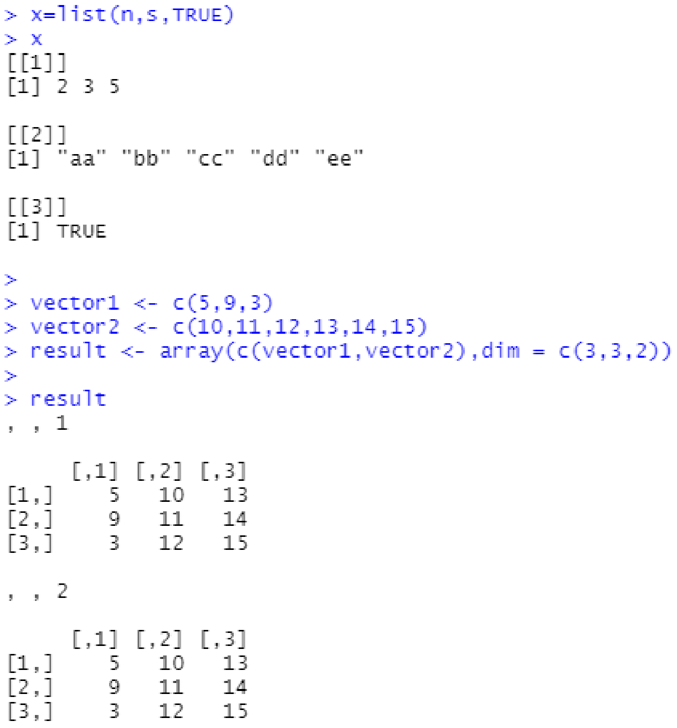
FOUNDATIONS FOR DATA ANALYTICS
LAB1
REPORT

MAJJIGA JASWANTH 20BCD7171

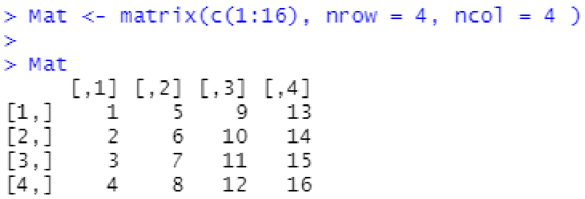
PRACTICE:



```
> x=5
> x
[1] 5
> x<-15
> x
[1] 15
> x<<-2
> x
[1] 2
> 25->x
> x
[1] 25
> x<-3
> x
[1] 3
> x!=2
[1] TRUE
> x<-2
> 2&3
[1] TRUE
> x<-2:8.
> x
[1] 2 3 4 5 6 7 8
> x<-2:8
> y<-5
> y%in%x
[1] TRUE
```

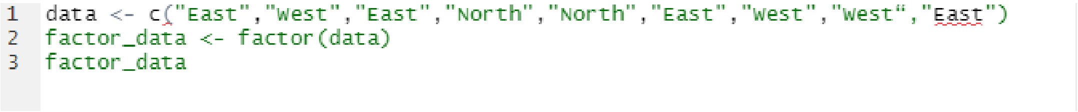


```
> x=list(n,s,TRUE)
> x
[[1]]
[1] 2 3 5

[[2]]
[1] "aa" "bb" "cc" "dd" "ee"

[[3]]
[1] TRUE

>
> vector1 <- c(5,9,3)
> vector2 <- c(10,11,12,13,14,15)
> result <- array(c(vector1,vector2),dim = c(3,3,2))
>
> result
, , 1

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15

, , 2

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15
```



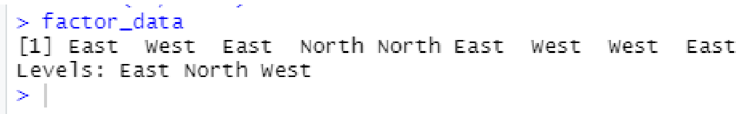
```
> Mat <- matrix(c(1:16), nrow = 4, ncol = 4 )
>
> Mat
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
```

pr1



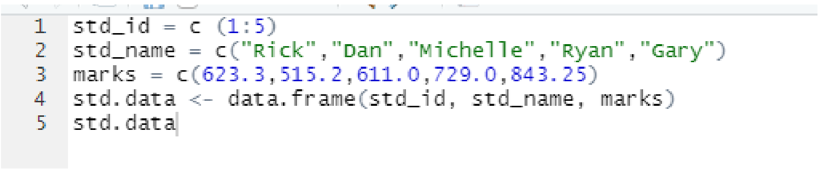
```
data <- c("East","West","East","North","North","East","West","West“,"East")
factor_data <- factor(data)
factor_data
```

output



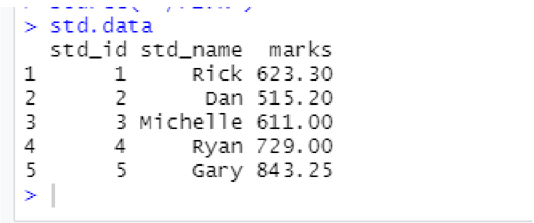
```
> factor_data
[1] East  West  East  North North East  West  West  East 
Levels: East North West
> 
```

pr2



```
std_id = c (1:5)
std_name = c("Rick","Dan","Michelle","Ryan","Gary")
marks = c(623.3,515.2,611.0,729.0,843.25)
std.data <- data.frame(std_id, std_name, marks)
std.data
```

output



```
> std.data
  std_id std_name  marks
1      1     Rick 623.30
2      2      Dan 515.20
3      3 Michelle 611.00
4      4     Ryan 729.00
5      5     Gary 843.25
> 
```

Exercise Questions:

1. Write a program in R to find the perfect numbers between 1 and 500.

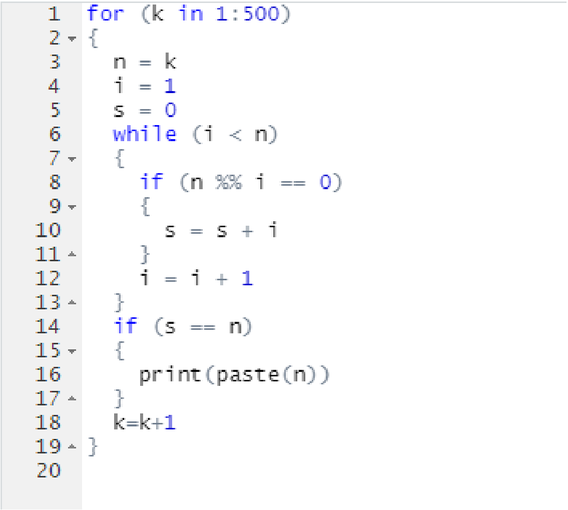
The perfect numbers between 1 to 500 are:

6

28

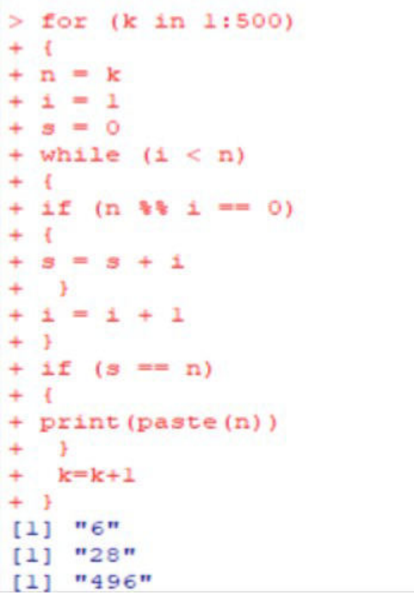
496

code:



```
for (k in 1:500)
{
  n = k
  i = 1
  s = 0
  while (i < n)
  {
    if (n %% i == 0)
    {
      s = s + i
    }
    i = i + 1
  }
  if (s == n)
  {
    print(paste(n))
  }
  k=k+1
}
```

output:



```
> for (k in 1:500)
+ {
+ n = k
+ i = 1
+ s = 0
+ while (i < n)
+ {
+ if (n %% i == 0)
+ {
+ s = s + i
+   }
+ i = i + 1
+ }
+ if (s == n)
+ {
+ print(paste(n))
+   }
+   k=k+1
+ }
[1] "6"
[1] "28"
[1] "496"
```

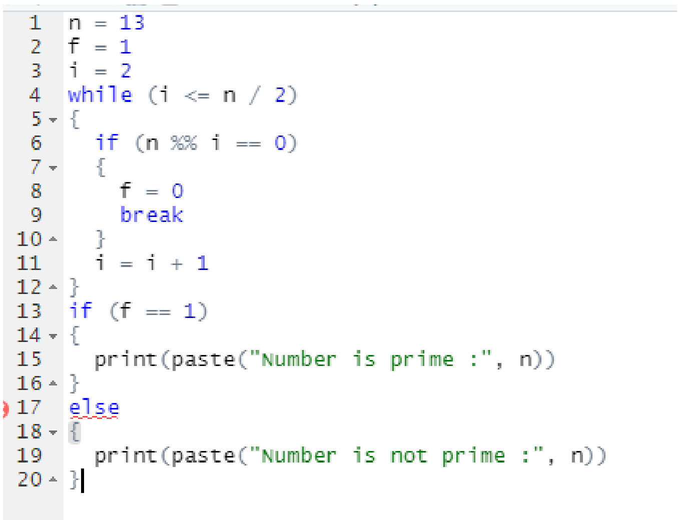
2. Write a program in R to check whether a number is prime or not.

Sample Output:

Input a number to check prime or not: 13

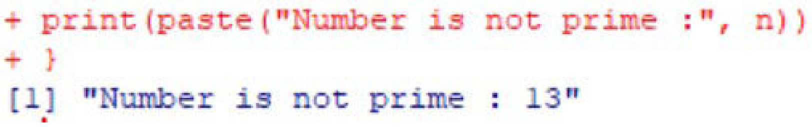
The entered number is a prime number.

Code:



```r
n = 13
f = 1
i = 2
while (i <= n / 2)
{
  if (n %% i == 0)
  {
    f = 0
    break
  }
  i = i + 1
}
if (f == 1)
{
  print(paste("Number is prime :", n))
}
else
{
  print(paste("Number is not prime :", n))
}
```

output:



```
+ print(paste("Number is not prime :", n))
+ }
[1] "Number is not prime : 13"
```

3. Write a program in R to find prime number within a range.

Input number for starting range: 1

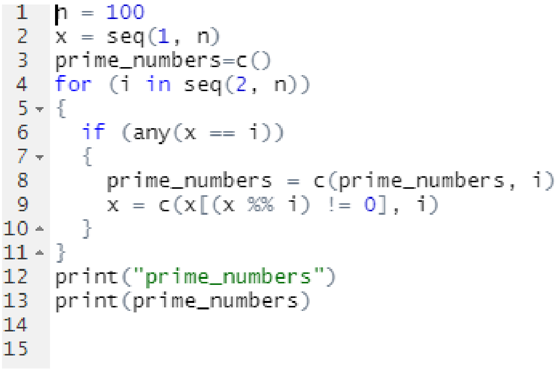
Input number for ending range: 100

The prime numbers between 1 and 100 are:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

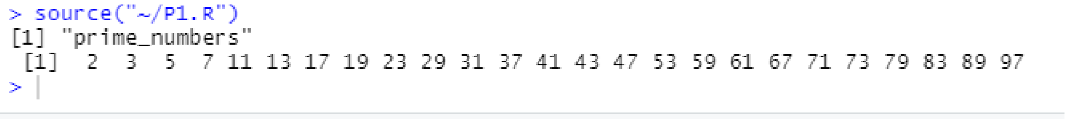
The total number of prime numbers between 1 to 100 is: 25

code:



```
n = 100
x = seq(1, n)
prime_numbers=c()
for (i in seq(2, n))
{
  if (any(x == i))
  {
    prime_numbers = c(prime_numbers, i)
    x = c(x[(x %% i) != 0], i)
  }
}
print("prime_numbers")
print(prime_numbers)


```

output:



```
> source("~/P1.R")
[1] "prime_numbers"
 [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
> 
```

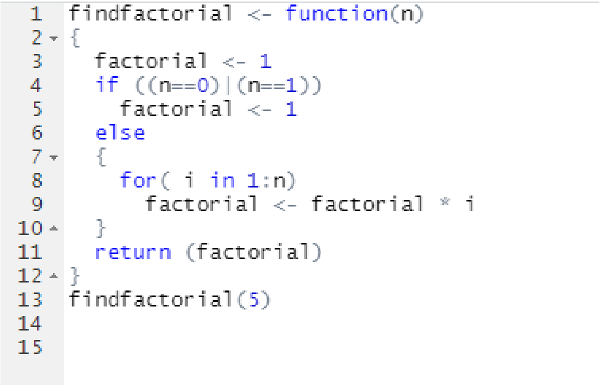
4. Write a program in R to find the factorial of a number.

Sample output:

Input a number to find the factorial: 5 The

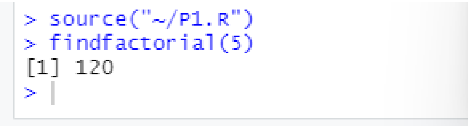
factorial of the given number is: 120

code:



```r
findfactorial <- function(n)
{
  factorial <- 1
  if ((n==0)|(n==1))
    factorial <- 1
  else
  {
    for( i in 1:n)
      factorial <- factorial * i
  }
  return (factorial)
}
findfactorial(5)
```

output:



```
> source("~/P1.R")
> findfactorial(5)
[1] 120
> 
```

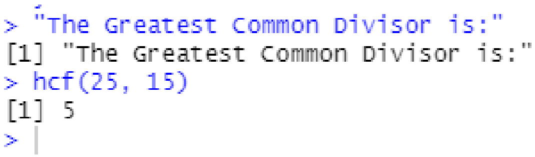
5. **Write a program in R to find the Greatest Common Divisor (GCD) of two numbers.**

Sample Output:

Input the first number: 25

Input the second number: 15 The Greatest Common Divisor is: 5

code:

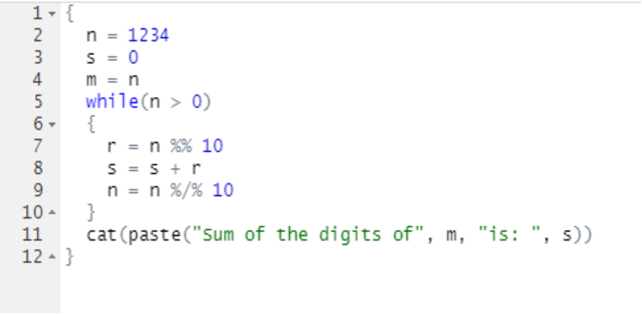


```
> "The Greatest Common Divisor is:"
[1] "The Greatest Common Divisor is:"
> hcf(25, 15)
[1] 5
> 
```

6. **Write a program in R to find the sum of digits of a given number. Sample Output: Input a number: 1234**

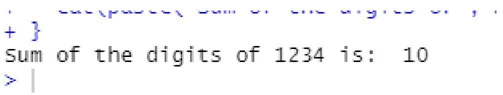
The sum of digits of 1234 is: 10

code:



```
{
  n = 1234
  s = 0
  m = n
  while(n > 0)
  {
    r = n %% 10
    s = s + r
    n = n %/% 10
  }
  cat(paste("Sum of the digits of", m, "is: ", s))
}
```

output:



```
+ }
Sum of the digits of 1234 is:  10
> 
```

7.Write a program in R to list non-prime numbers from 1 to an upperbound.

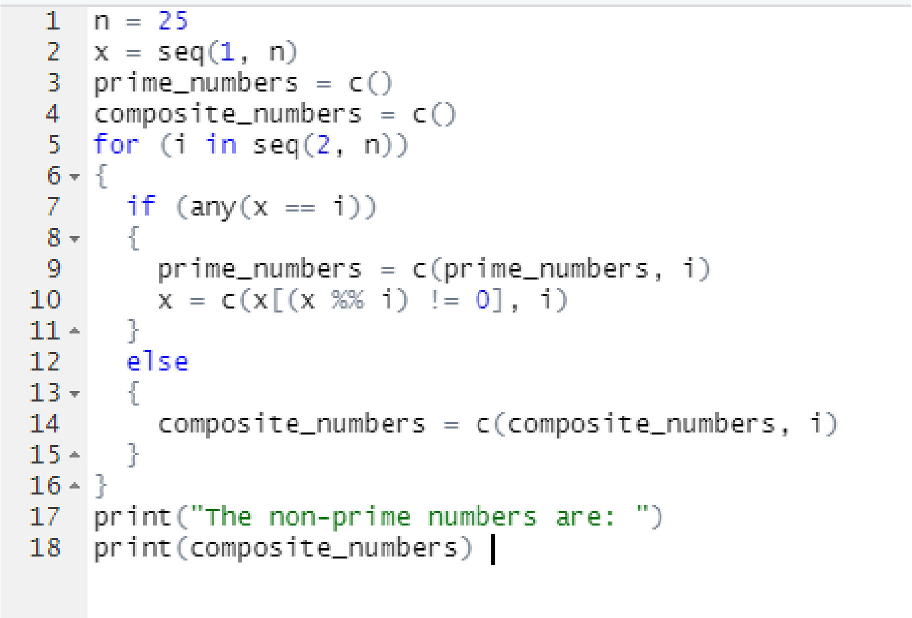
Sample Output:

Input the upperlimit: 25

The non-prime numbers are:

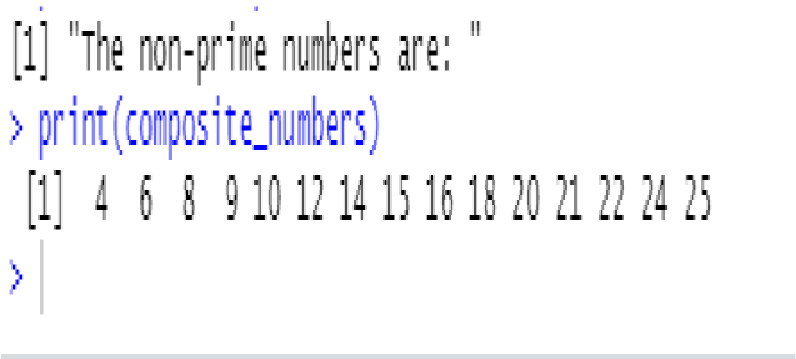
4 6 8 9 10 12 14 15 16 18 20 21 22 24 25

code:



```
n = 25
x = seq(1, n)
prime_numbers = c()
composite_numbers = c()
for (i in seq(2, n))
{
  if (any(x == i))
  {
    prime_numbers = c(prime_numbers, i)
    x = c(x[(x %% i) != 0], i)
  }
  else
  {
    composite_numbers = c(composite_numbers, i)
  }
}
print("The non-prime numbers are: ")
print(composite_numbers)
```

output:



```
[1] "The non-prime numbers are: "
> print(composite_numbers)
 [1]  4  6  8  9 10 12 14 15 16 18 20 21 22 24 25
> 
```

8. Write a program in R to print a square pattern with # character.

Sample Output:

Print a pattern like square with # character:

--

Input the number of characters for a side: 4

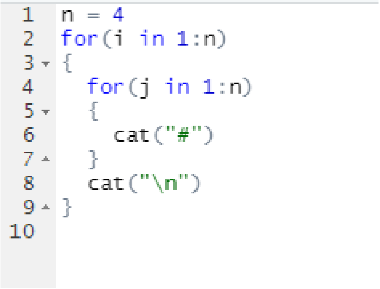
# # # #

# # # #

# # # #

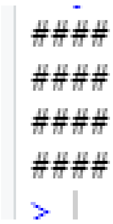
# # # #

code:



```
n = 4
for(i in 1:n)
{
  for(j in 1:n)
  {
    cat("#")
  }
  cat("\n")
}
```

output:



```
####
####
####
####
>
```

9. Write a program in R to display the

cube of the number upto given an integer.

Sample Output:

Input the number of terms : 5

Number is : 1 and the cube of 1 is: 1

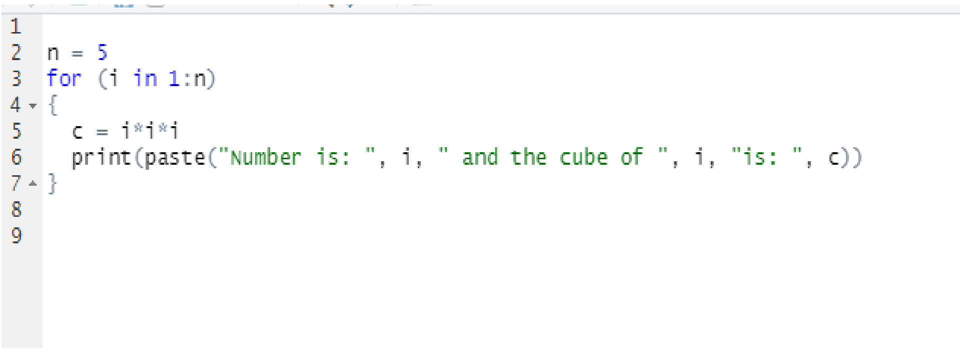
Number is : 2 and the cube of 2 is: 8

Number is : 3 and the cube of 3 is: 27

Number is : 4 and the cube of 4 is: 64

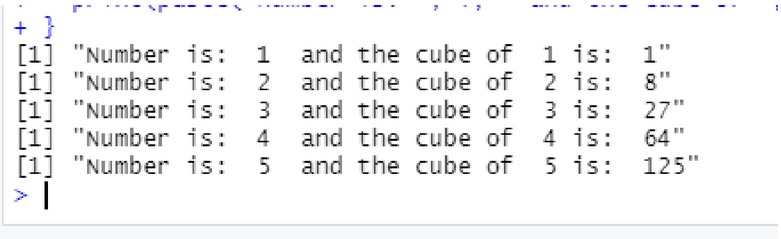
Number is : 5 and the cube of 5 is: 125

code:



```

n = 5
for (i in 1:n)
{
  c = i*i*i
  print(paste("Number is: ", i, " and the cube of ", i, "is: ", c))
}


```

output:



```
+ }
[1] "Number is:  1  and the cube of  1 is:  1"
[1] "Number is:  2  and the cube of  2 is:  8"
[1] "Number is:  3  and the cube of  3 is:  27"
[1] "Number is:  4  and the cube of  4 is:  64"
[1] "Number is:  5  and the cube of  5 is:  125"
> 
```

10. Write a program in R to display the

first n terms of Fibonacci series.

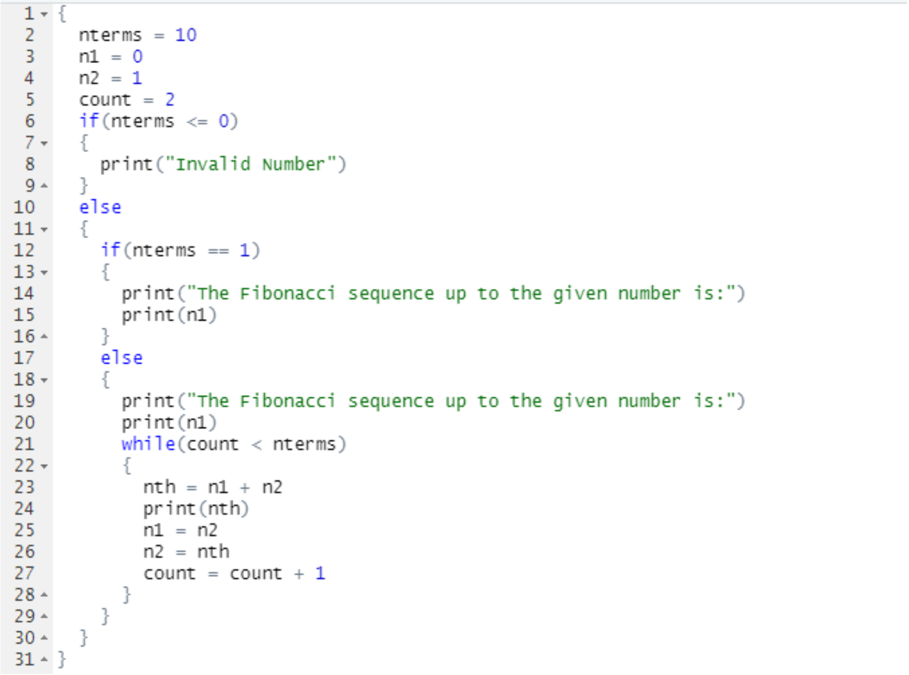
Sample Output:

Input number of terms to display: 10

Here is the Fibonacci series upto to 10 terms:

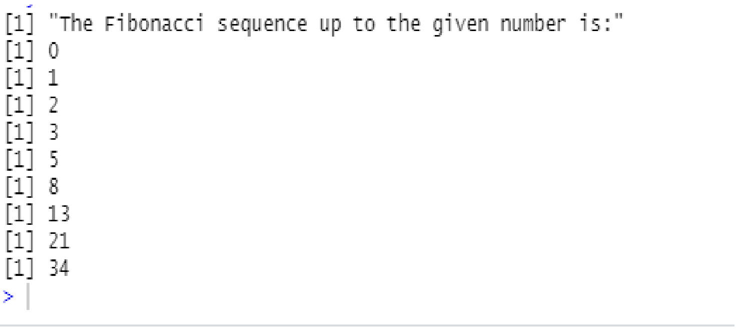
0 1 1 2 3 5 8 13 21 34

code:



```
{
  nterms = 10
  n1 = 0
  n2 = 1
  count = 2
  if(nterms <= 0)
  {
    print("Invalid Number")
  }
  else
  {
    if(nterms == 1)
    {
      print("The Fibonacci sequence up to the given number is:")
      print(n1)
    }
    else
    {
      print("The Fibonacci sequence up to the given number is:")
      print(n1)
      while(count < nterms)
      {
        nth = n1 + n2
        print(nth)
        n1 = n2
        n2 = nth
        count = count + 1
      }
    }
  }
}
```

output:



```
[1] "The Fibonacci sequence up to the given number is:"
[1] 0
[1] 1
[1] 2
[1] 3
[1] 5
[1] 8
[1] 13
[1] 21
[1] 34
>
```

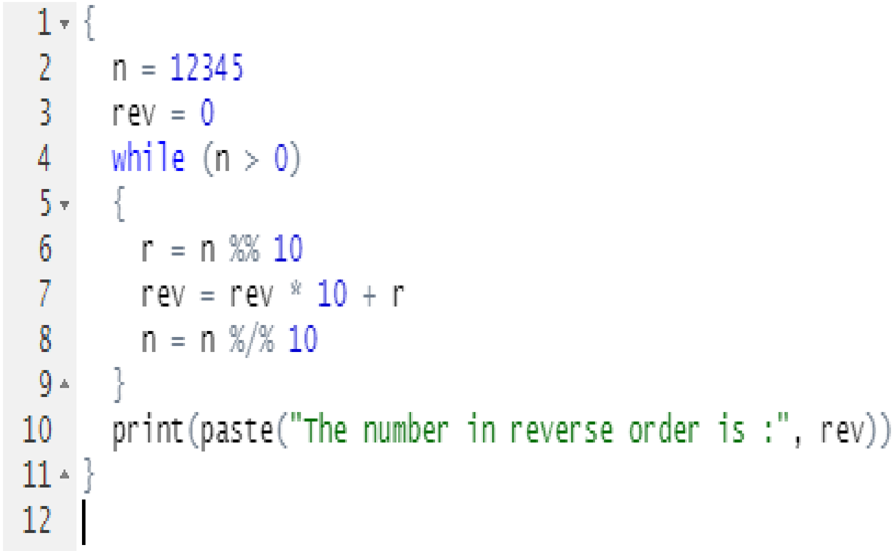
11. Write a program in R to display the number in reverse order.

Sample Output:

Input a number: 12345

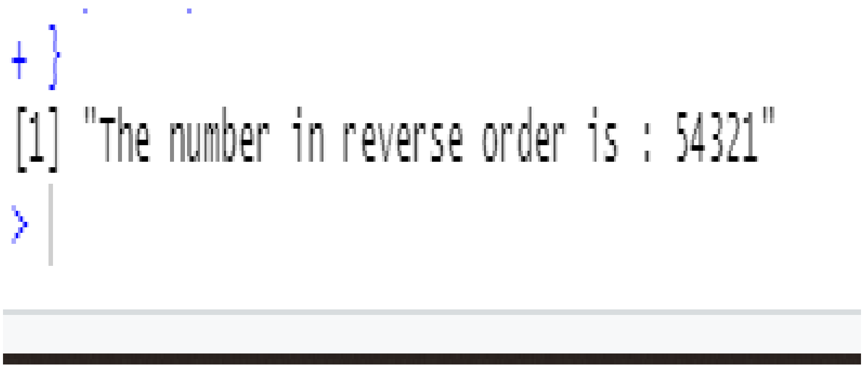
The number in reverse order is : 54321

code:



```
{
  n = 12345
  rev = 0
  while (n > 0)
  {
    r = n %% 10
    rev = rev * 10 + r
    n = n %/% 10
  }
  print(paste("The number in reverse order is :", rev))
}

```

output:



```
+ }
[1] "The number in reverse order is : 54321"
> 
```

12. Write a program in R to find out the sum of an A.P. series.

Sample Output:

Input the starting number of the A.P. series: 1

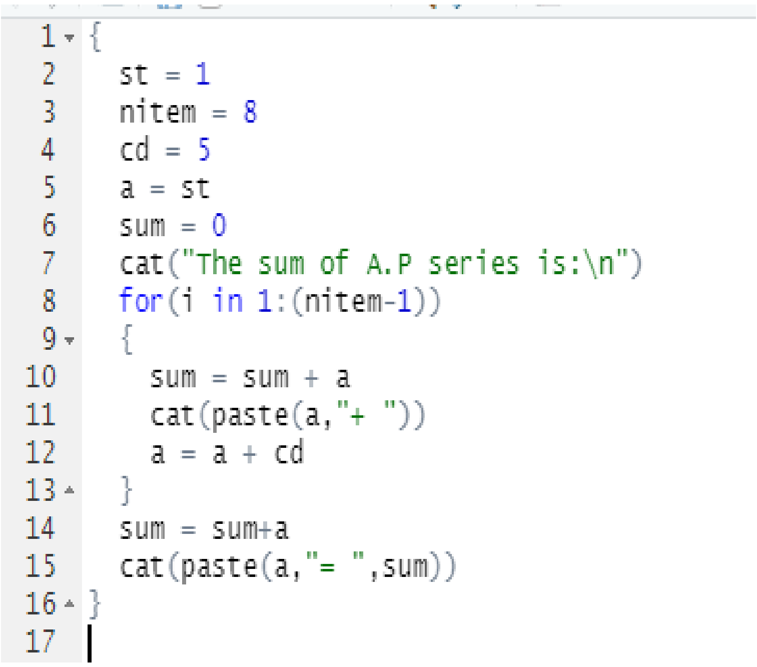
Input the number of items for the A.P. series: 8

Input the common difference of A.P. series: 5 The

Sum of the A.P. series are :

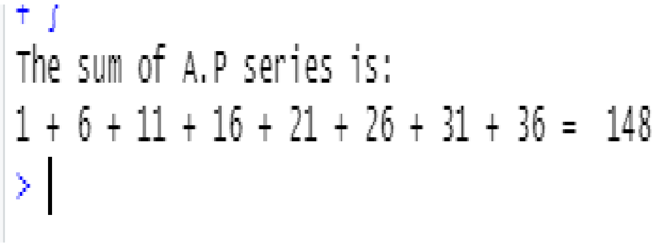
1 + 6 + 11 + 16 + 21 + 26 + 31 + 36 = 148

code:



```r
{
  st = 1
  nitem = 8
  cd = 5
  a = st
  sum = 0
  cat("The sum of A.P series is:\n")
  for(i in 1:(nitem-1))
  {
    sum = sum + a
    cat(paste(a,"+ "))
    a = a + cd
  }
  sum = sum+a
  cat(paste(a,"= ",sum))
}
```

output:



```
+ }
The sum of A.P series is:
1 + 6 + 11 + 16 + 21 + 26 + 31 + 36 =  148
> 
```

13. Write a program in R to Check

Whether a Number can be Express as Sum

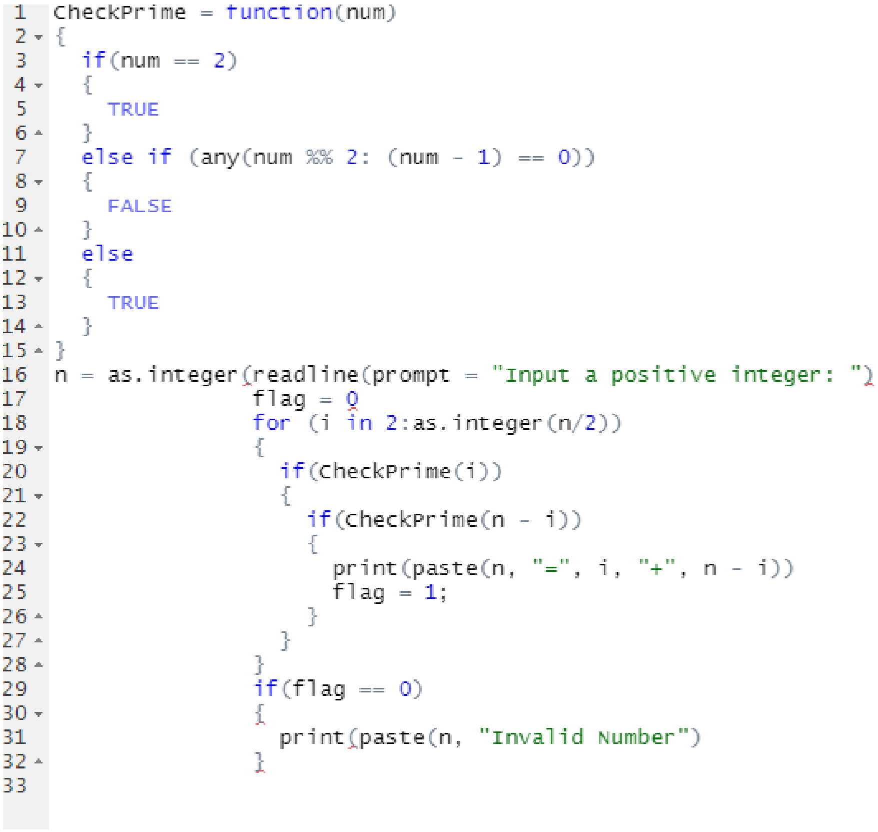
of Two Prime Numbers.

Sample Output:

Input a positive integer: 20

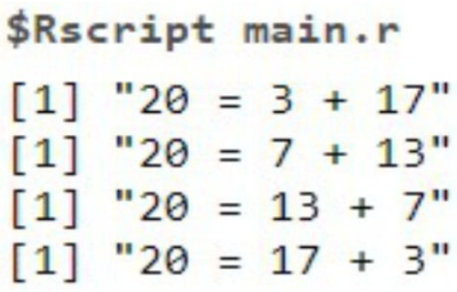
20 = 3 + 17

20 = 7 + 13



```
CheckPrime = function(num)
{
  if(num == 2)
  {
    TRUE
  }
  else if (any(num %% 2: (num - 1) == 0))
  {
    FALSE
  }
  else
  {
    TRUE
  }
}
n = as.integer(readline(prompt = "Input a positive integer: ")
               flag = 0
               for (i in 2:as.integer(n/2))
               {
                 if(CheckPrime(i))
                 {
                   if(CheckPrime(n - i))
                   {
                     print(paste(n, "=", i, "+", n - i))
                     flag = 1;
                   }
                 }
               }
               if(flag == 0)
               {
                 print(paste(n, "Invalid Number")
               }
```

output:



```
$Rscript main.r

[1] "20 = 3 + 17"
[1] "20 = 7 + 13"
[1] "20 = 13 + 7"
[1] "20 = 17 + 3"
```

14.Write a program in R to find the length of a string without using the library function.

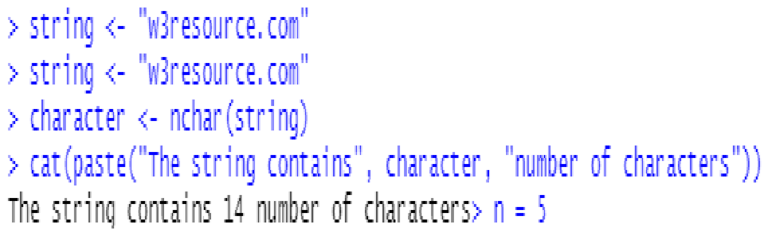
Sample Output:

Input a string: w3resource.com

The string contains 14 number of characters.

So, the length of the string w3resource.com is:14

input:



```
> string <- "w3resource.com"
> string <- "w3resource.com"
> character <- nchar(string)
> cat(paste("The string contains", character, "number of characters"))
The string contains 14 number of characters> n = 5
```

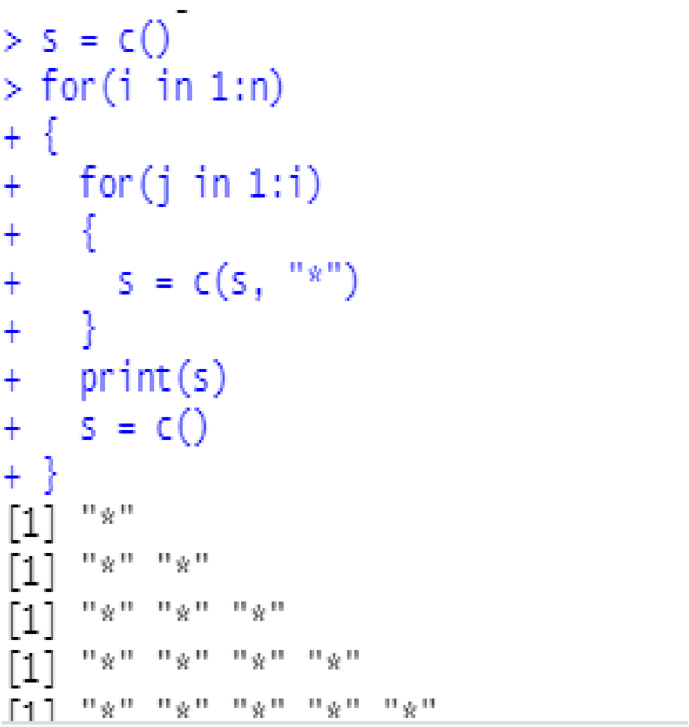
15.Write a program in R to display the pattern like right angle triangle using an asterisk.

Sample Output:

Input number of rows: 5

```
*
**
***
****
*****
```

code:



```
> s = c()
> for(i in 1:n)
+ {
+   for(j in 1:i)
+   {
+     s = c(s, "*")
+   }
+   print(s)
+   s = c()
+ }
[1] "*"
[1] "*" "*"
[1] "*" "*" "*"
[1] "*" "*" "*" "*"
[1] "*" "*" "*" "*" "*"
```

16. Write a program in R to display the pattern like right angle triangle with number.

Sample Output:

Input number of rows: 5

1

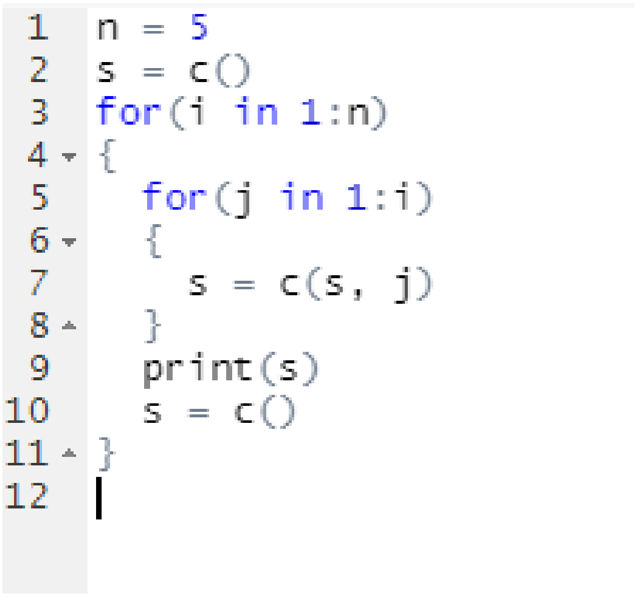
12

123

1234

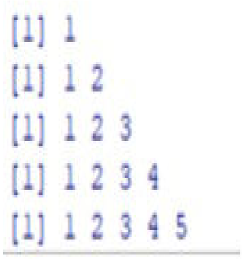
12345

code:



```
n = 5
s = c()
for(i in 1:n)
{
  for(j in 1:i)
  {
    s = c(s, j)
  }
  print(s)
  s = c()
}

```

output:



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

17. Write a program in R to make such a pattern like right angle triangle using number which will repeat the number for that row.

Sample Output:

Input number of rows: 5

1

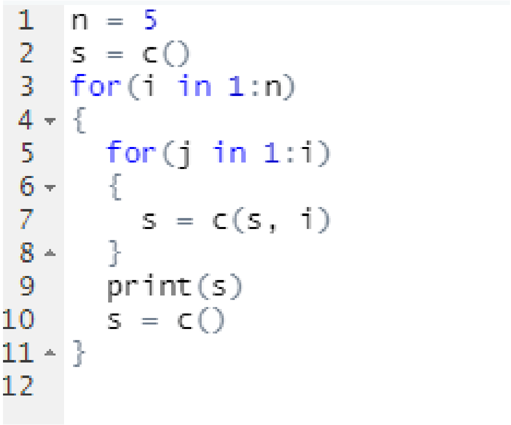
22

333

4444

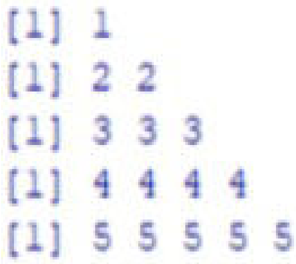
55555

code:



```
n = 5
s = c()
for(i in 1:n)
{
  for(j in 1:i)
  {
    s = c(s, i)
  }
  print(s)
  s = c()
}

```

output:



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

18. Write a program in R to make such a pattern like right angle triangle with number increased by 1.

Sample Output:

Input number of rows: 4

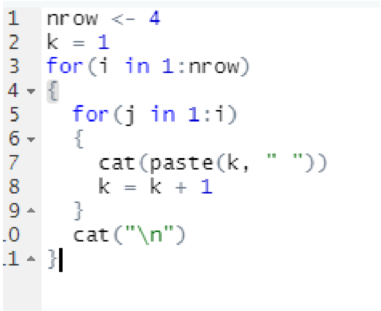
1

2 3

4 5 6

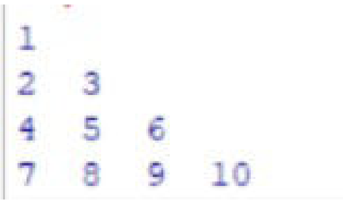
7 8 9 10

code:



```
nrow <- 4
k = 1
for(i in 1:nrow)
{
  for(j in 1:i)
  {
    cat(paste(k, " "))
    k = k + 1
  }
  cat("\n")
}
```

output:



```
1
2  3
4  5  6
7  8  9  10
```

19. Write a program in R to find the sum of first and last digit of a number.

Sample Output:

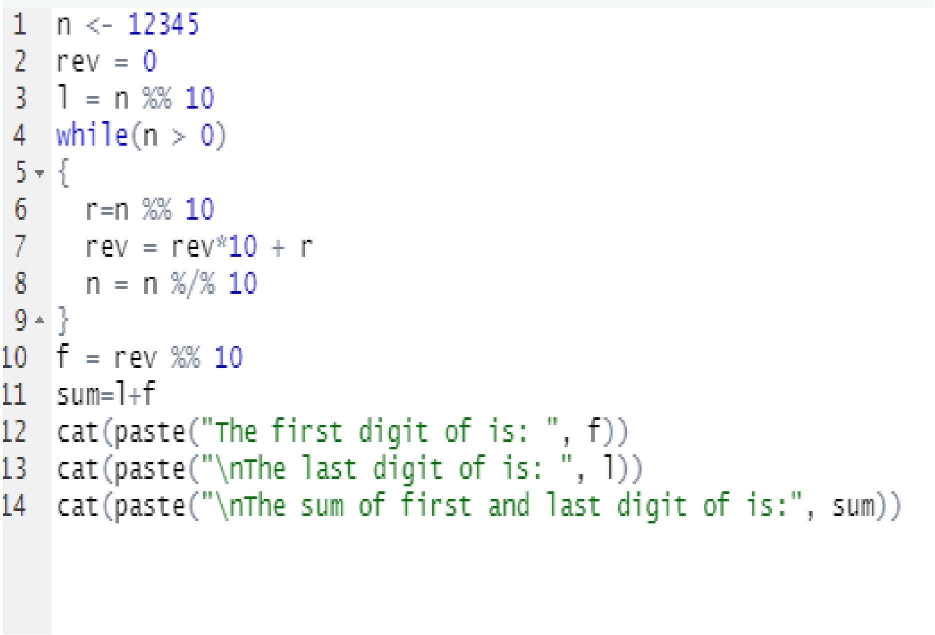
Input any number: 12345

The first digit of 12345 is: 1

The last digit of 12345 is: 5

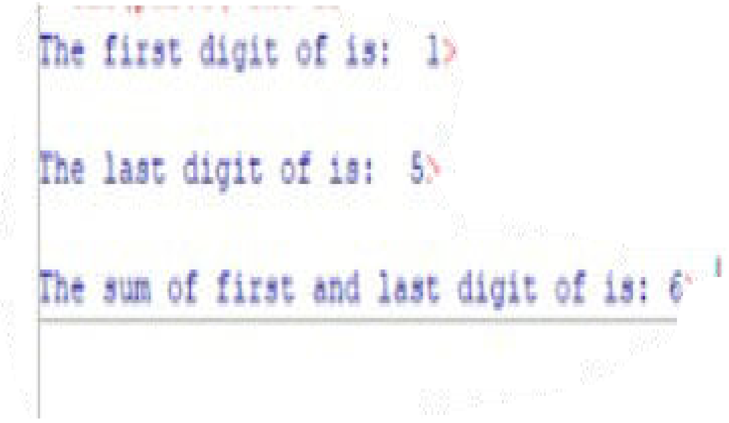
The sum of first and last digit of 12345 is: 6

code:



```
n <- 12345
rev = 0
l = n %% 10
while(n > 0)
{
  r=n %% 10
  rev = rev*10 + r
  n = n %/% 10
}
f = rev %% 10
sum=l+f
cat(paste("The first digit of is: ", f))
cat(paste("\nThe last digit of is: ", l))
cat(paste("\nThe sum of first and last digit of is:", sum))
```

output:



```
The first digit of is:  1

The last digit of is:  5

The sum of first and last digit of is: 6
```

20. Write a program in R to find the frequency of each digit in a given integer.

Sample Output:

Input any number: 122345

The frequency of 0 = 0

The frequency of 1 = 1

The frequency of 2 = 2

The frequency of 3 = 1

The frequency of 4 = 1

The frequency of 5 = 1

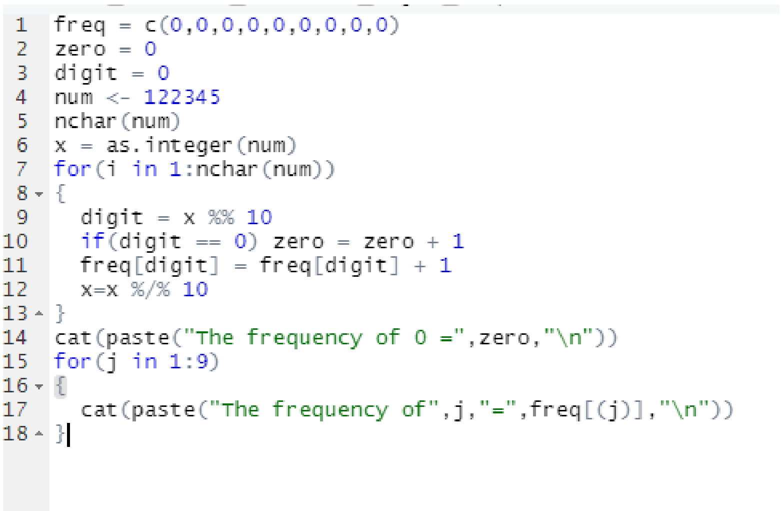
The frequency of 6 = 0

The frequency of 7 = 0

The frequency of 8 = 0

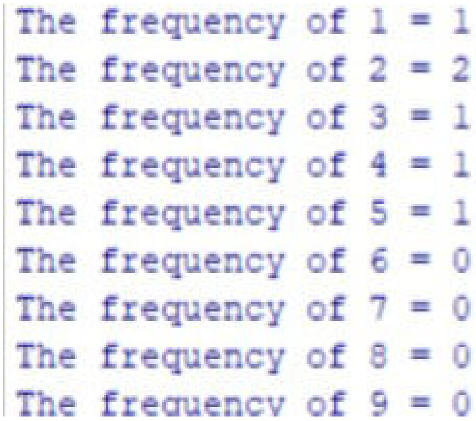
The frequency of 9 = 0

code:



```r
freq = c(0,0,0,0,0,0,0,0,0)
zero = 0
digit = 0
num <- 122345
nchar(num)
x = as.integer(num)
for(i in 1:nchar(num))
{
  digit = x %% 10
  if(digit == 0) zero = zero + 1
  freq[digit] = freq[digit] + 1
  x=x %/% 10
}
cat(paste("The frequency of 0 =",zero,"\n"))
for(j in 1:9)
{
  cat(paste("The frequency of",j,"=",freq[(j)],"\n"))
}
```

output:



```
The frequency of 1 = 1
The frequency of 2 = 2
The frequency of 3 = 1
The frequency of 4 = 1
The frequency of 5 = 1
The frequency of 6 = 0
The frequency of 7 = 0
The frequency of 8 = 0
The frequency of 9 = 0
```

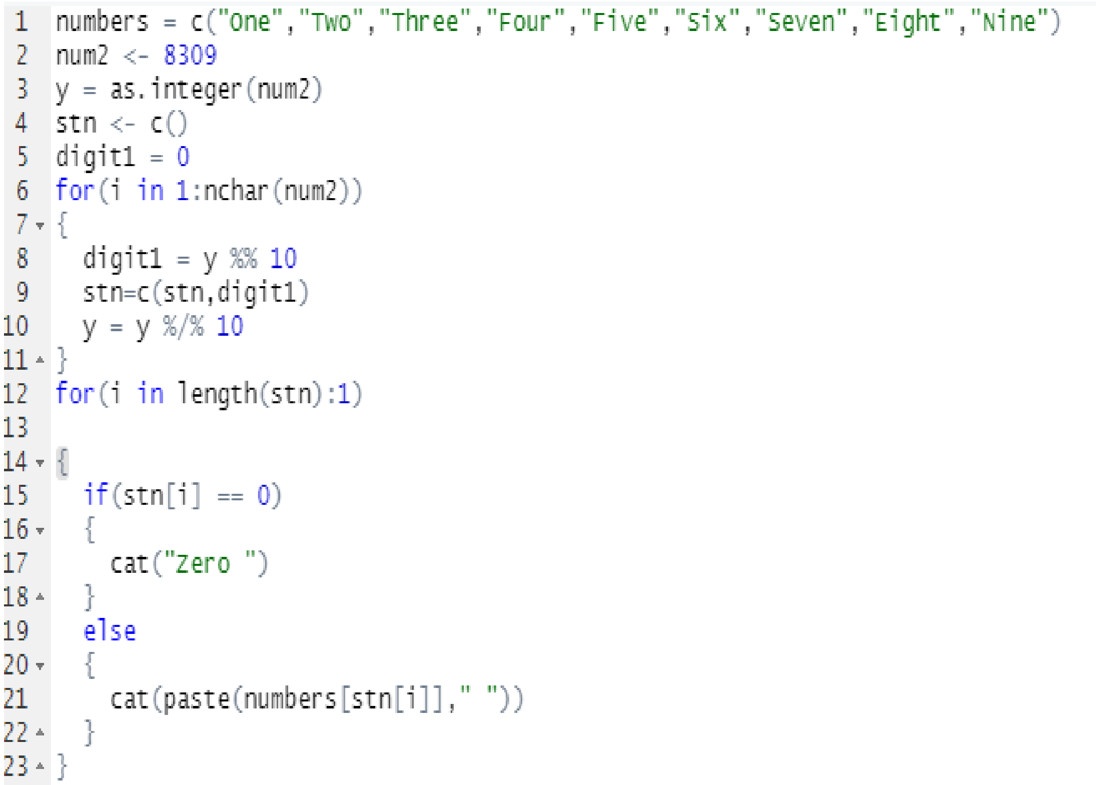
21. Write a program in R to display the given number in words.

Sample Output:

Input any number: 8309

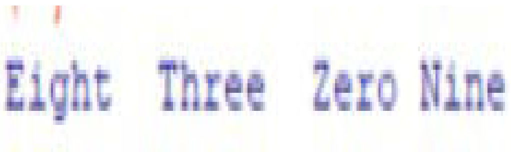
Eight Three Zero Nine

code:



```r
numbers = c("One","Two","Three","Four","Five","Six","Seven","Eight","Nine")
num2 <- 8309
y = as.integer(num2)
stn <- c()
digit1 = 0
for(i in 1:nchar(num2))
{
  digit1 = y %% 10
  stn=c(stn,digit1)
  y = y %/% 10
}
for(i in length(stn):1)

{
  if(stn[i] == 0)
  {
    cat("Zero ")
  }
  else
  {
    cat(paste(numbers[stn[i]]," "))
  }
}
```

output:



```
Eight  Three  Zero Nine
```

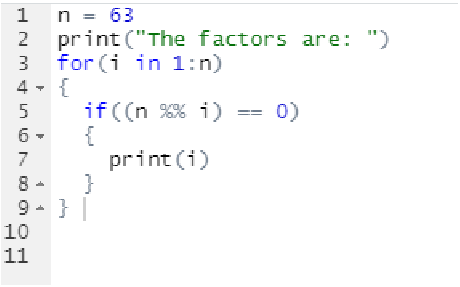
22. Write a program in R to enter any number and print all factors of the number.

Sample Output:

Input a number: 63

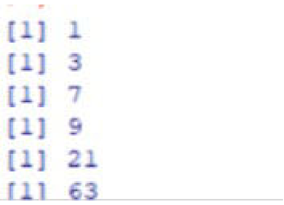
The factors are: 1 3 7 9 21 63

code:



```
n = 63
print("The factors are: ")
for(i in 1:n)
{
  if((n %% i) == 0)
  {
    print(i)
  }
}
```

output:



```
[1] 1
[1] 3
[1] 7
[1] 9
[1] 21
[1] 63
```

23. Write a program in R to find one's complement of a binary number.

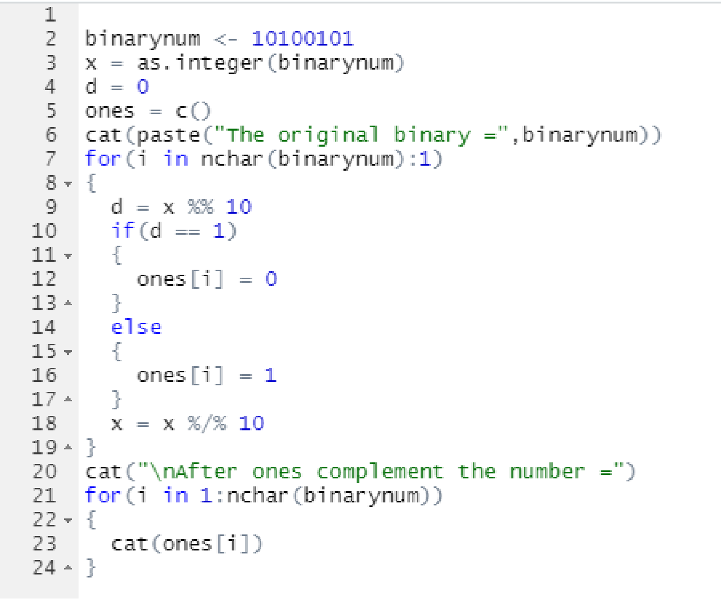
Sample Output:

Input a 8 bit binary value: 10100101

The original binary = 10100101

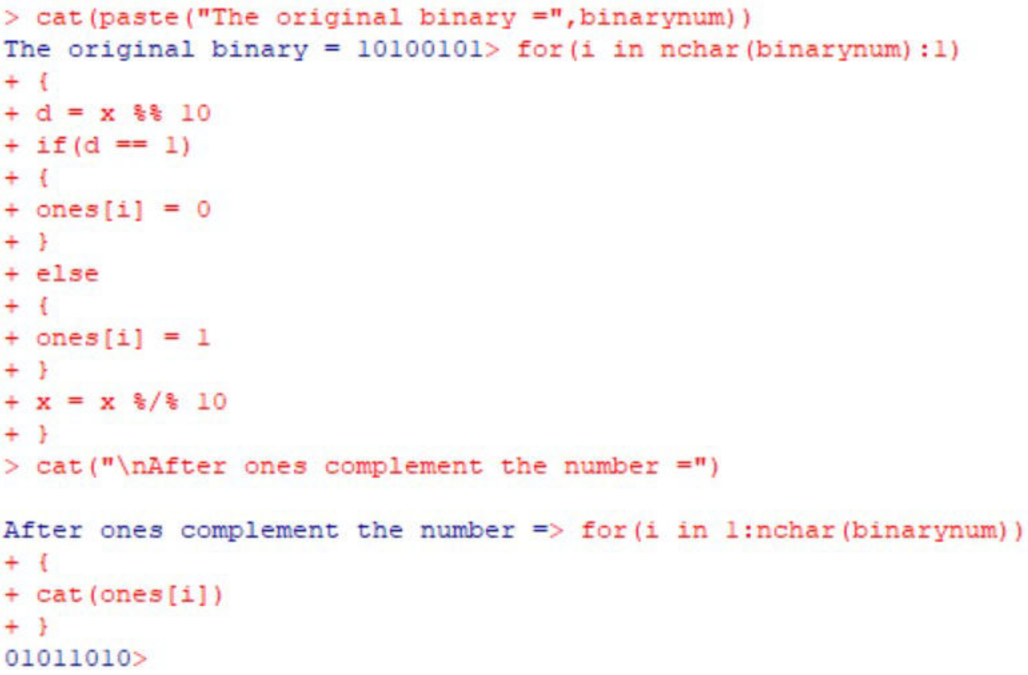
After ones complement the number = 01011010

code:



```r

binarynum <- 10100101
x = as.integer(binarynum)
d = 0
ones = c()
cat(paste("The original binary =",binarynum))
for(i in nchar(binarynum):1)
{
  d = x %% 10
  if(d == 1)
  {
    ones[i] = 0
  }
  else
  {
    ones[i] = 1
  }
  x = x %/% 10
}
cat("\nAfter ones complement the number =")
for(i in 1:nchar(binarynum))
{
  cat(ones[i])
}
```

output:



```
> cat(paste("The original binary =",binarynum))
The original binary = 10100101> for(i in nchar(binarynum):1)
+ {
+ d = x %% 10
+ if(d == 1)
+ {
+ ones[i] = 0
+ }
+ else
+ {
+ ones[i] = 1
+ }
+ x = x %/% 10
+ }
> cat("\nAfter ones complement the number =")

After ones complement the number => for(i in 1:nchar(binarynum))
+ {
+ cat(ones[i])
+ }
01011010>
```

24: ----- DIDNT GET IT

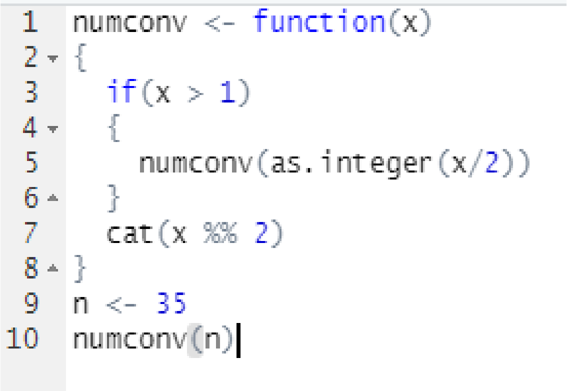
25.Write a program in R to convert a decimal number to binary number.

Sample Output:

Input a decimal number: 35

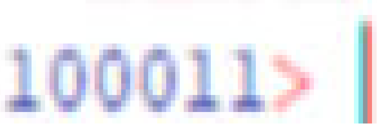
The binary number is: 100011

CODE:



```
numconv <- function(x)
{
  if(x > 1)
  {
    numconv(as.integer(x/2))
  }
  cat(x %% 2)
}
n <- 35
numconv(n)
```

output:



```
100011>
```

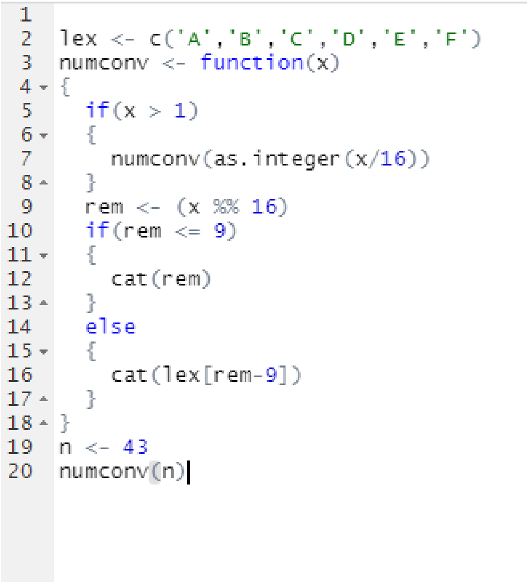
26. Write a program in R to convert a decimal number to hexadecimal number.

Sample Output:

Input a decimal number: 43

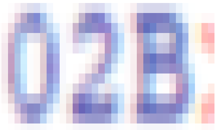
The hexadecimal number is : 2B

code:



```

lex <- c('A','B','C','D','E','F')
numconv <- function(x)
{
  if(x > 1)
  {
    numconv(as.integer(x/16))
  }
  rem <- (x %% 16)
  if(rem <= 9)
  {
    cat(rem)
  }
  else
  {
    cat(lex[rem-9])
  }
}
n <- 43
numconv(n)
```

output:



```
02B
```

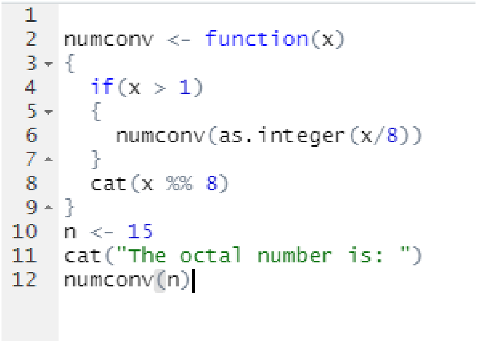
27. Write a program in R to convert a decimal number to octal number.

Sample Output:

Input a decimal number: 15

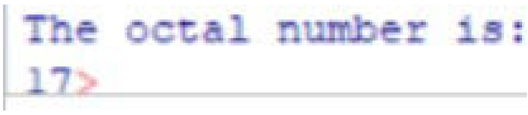
The octal number is: 17

code:



```

numconv <- function(x)
{
  if(x > 1)
  {
    numconv(as.integer(x/8))
  }
  cat(x %% 8)
}
n <- 15
cat("The octal number is: ")
numconv(n)
```

output:



```
The octal number is:
17>
```

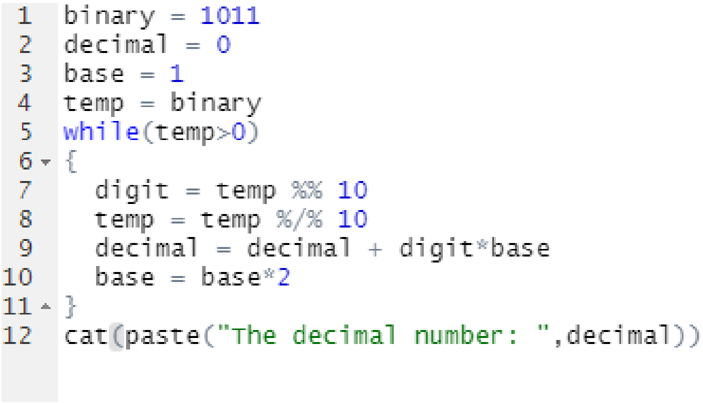
28. Write a program in R to convert a binary number to decimal number.

Sample Output:

Input a binary number: 1011

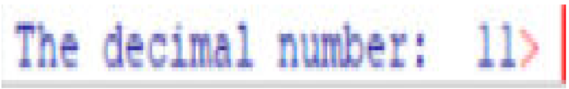
The decimal number: 11

code:



```
binary = 1011
decimal = 0
base = 1
temp = binary
while(temp>0)
{
  digit = temp %% 10
  temp = temp %/% 10
  decimal = decimal + digit*base
  base = base*2
}
cat(paste("The decimal number: ",decimal))
```

output:



```
The decimal number:  11>
```

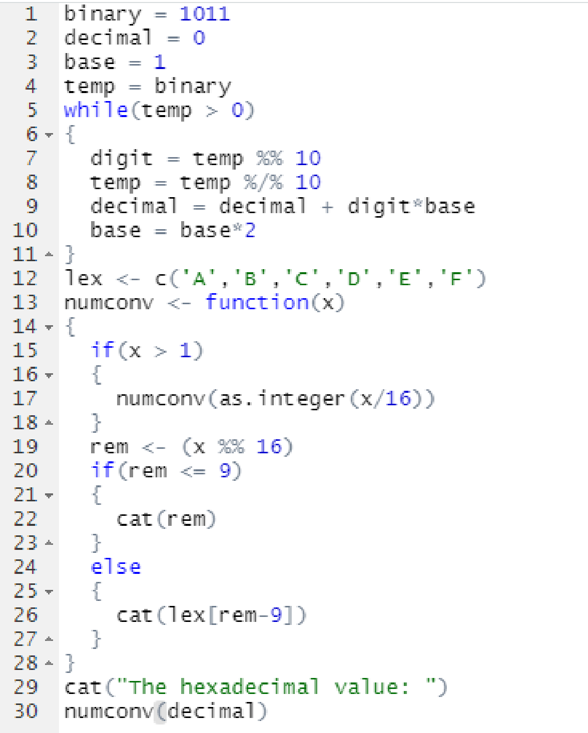
29. Write a program in R to convert a binary number to hexadecimal number.

Sample Output:

Input a binary number: 1011

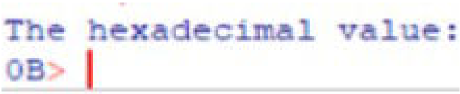
The hexadecimal value: B

code:



```
binary = 1011
decimal = 0
base = 1
temp = binary
while(temp > 0)
{
  digit = temp %% 10
  temp = temp %/% 10
  decimal = decimal + digit*base
  base = base*2
}
lex <- c('A','B','C','D','E','F')
numconv <- function(x)
{
  if(x > 1)
  {
    numconv(as.integer(x/16))
  }
  rem <- (x %% 16)
  if(rem <= 9)
  {
    cat(rem)
  }
  else
  {
    cat(lex[rem-9])
  }
}
cat("The hexadecimal value: ")
numconv(decimal)
```

output:



```
The hexadecimal value:
0B>
```

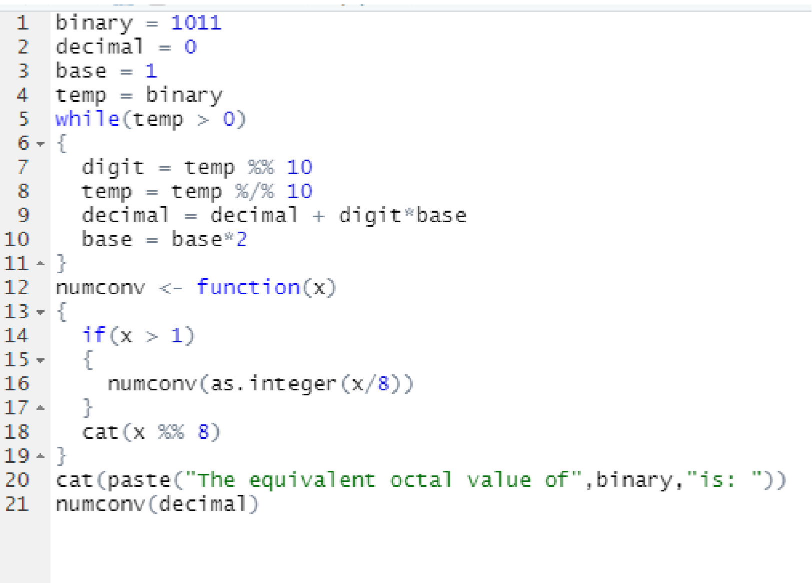
30. Write a program in R to convert a binary number to hexadecimal number.

Sample Output:

Input a binary number: 1011

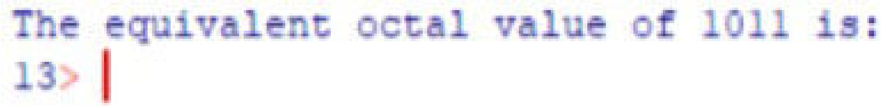
The equivalent octal value of 1011 is : 13

code:



```r
binary = 1011
decimal = 0
base = 1
temp = binary
while(temp > 0)
{
  digit = temp %% 10
  temp = temp %/% 10
  decimal = decimal + digit*base
  base = base*2
}
numconv <- function(x)
{
  if(x > 1)
  {
    numconv(as.integer(x/8))
  }
  cat(x %% 8)
}
cat(paste("The equivalent octal value of",binary,"is: "))
numconv(decimal)
```

output:



```
The equivalent octal value of 1011 is:
13>
```