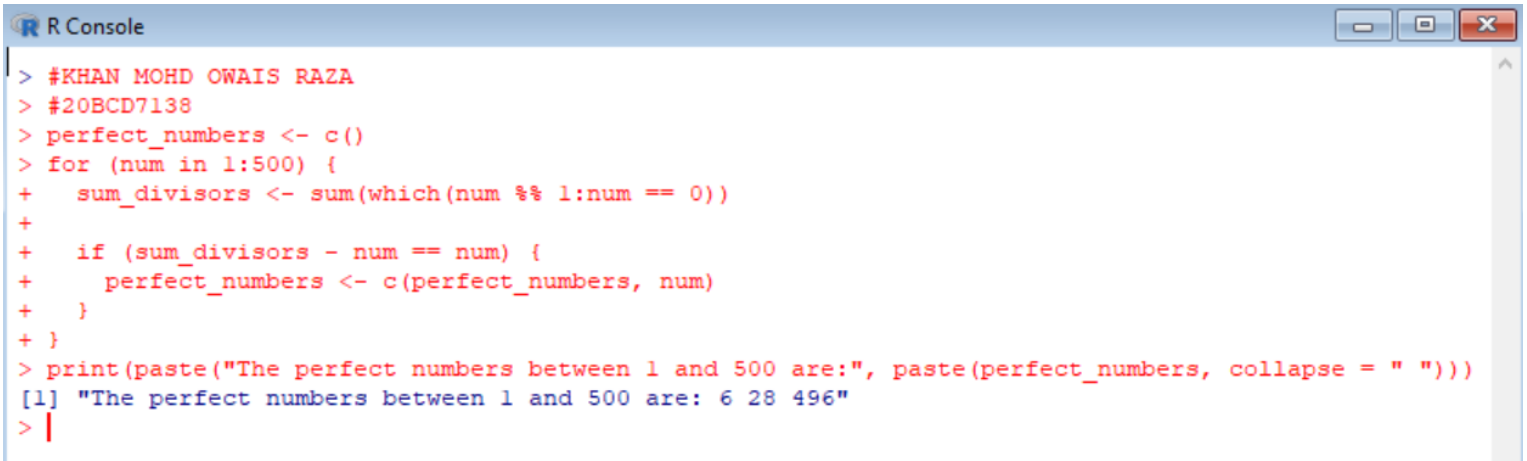
FDA Lab-2

KHAN MOHD OWAIS RAZA
20BCD7138

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

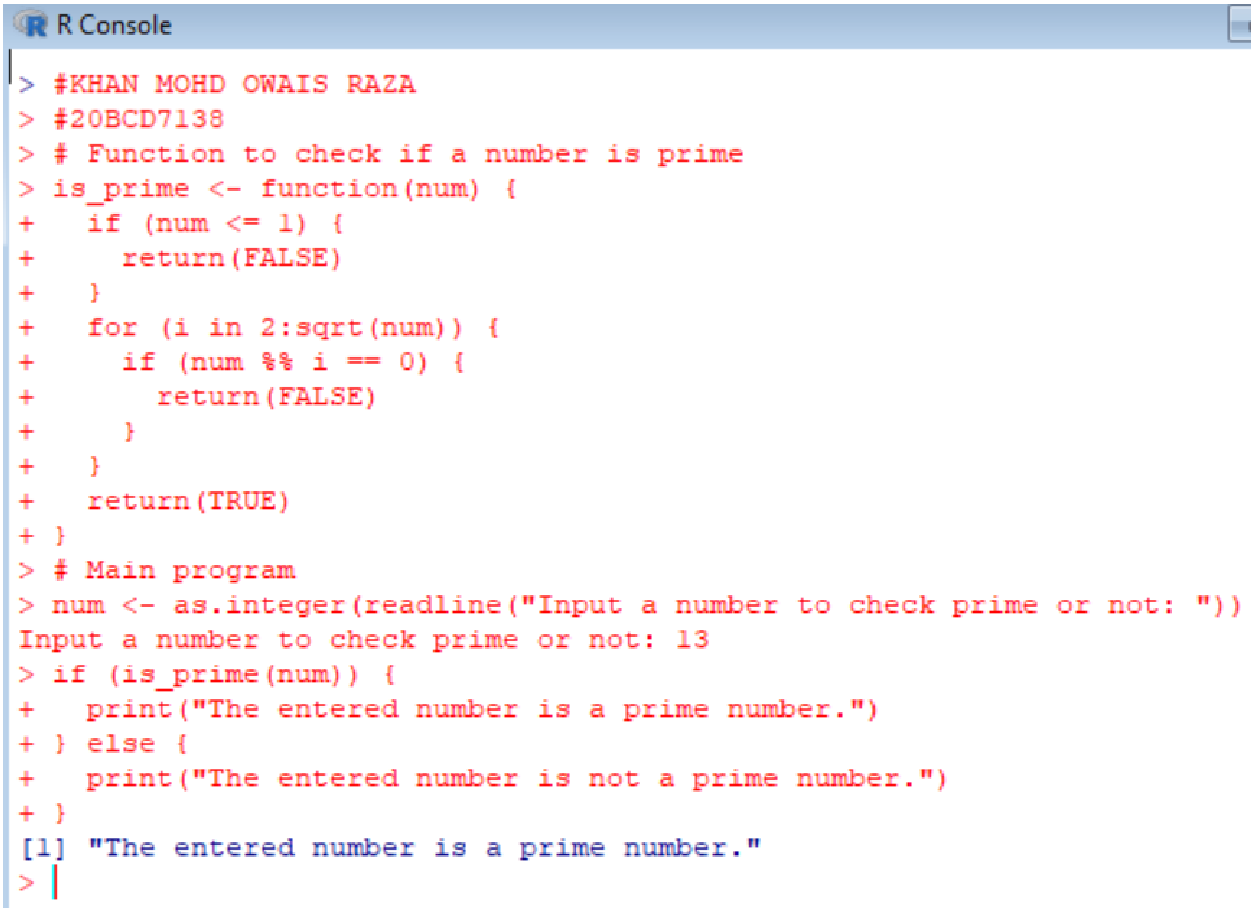
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
perfect_numbers <- c()
for (num in 1:500) {
  sum_divisors <- sum(which(num %% 1:num == 0))

  if (sum_divisors - num == num) {
    perfect_numbers <- c(perfect_numbers, num)
  }
}
print(paste("The perfect numbers between 1 and 500 are:",
paste(perfect_numbers, collapse = " ")))
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> perfect_numbers <- c()
> for (num in 1:500) {
+   sum_divisors <- sum(which(num %% 1:num == 0))
+
+   if (sum_divisors - num == num) {
+     perfect_numbers <- c(perfect_numbers, num)
+   }
+ }
> print(paste("The perfect numbers between 1 and 500 are:", paste(perfect_numbers, collapse = " ")))
[1] "The perfect numbers between 1 and 500 are: 6 28 496"
> 
```

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.

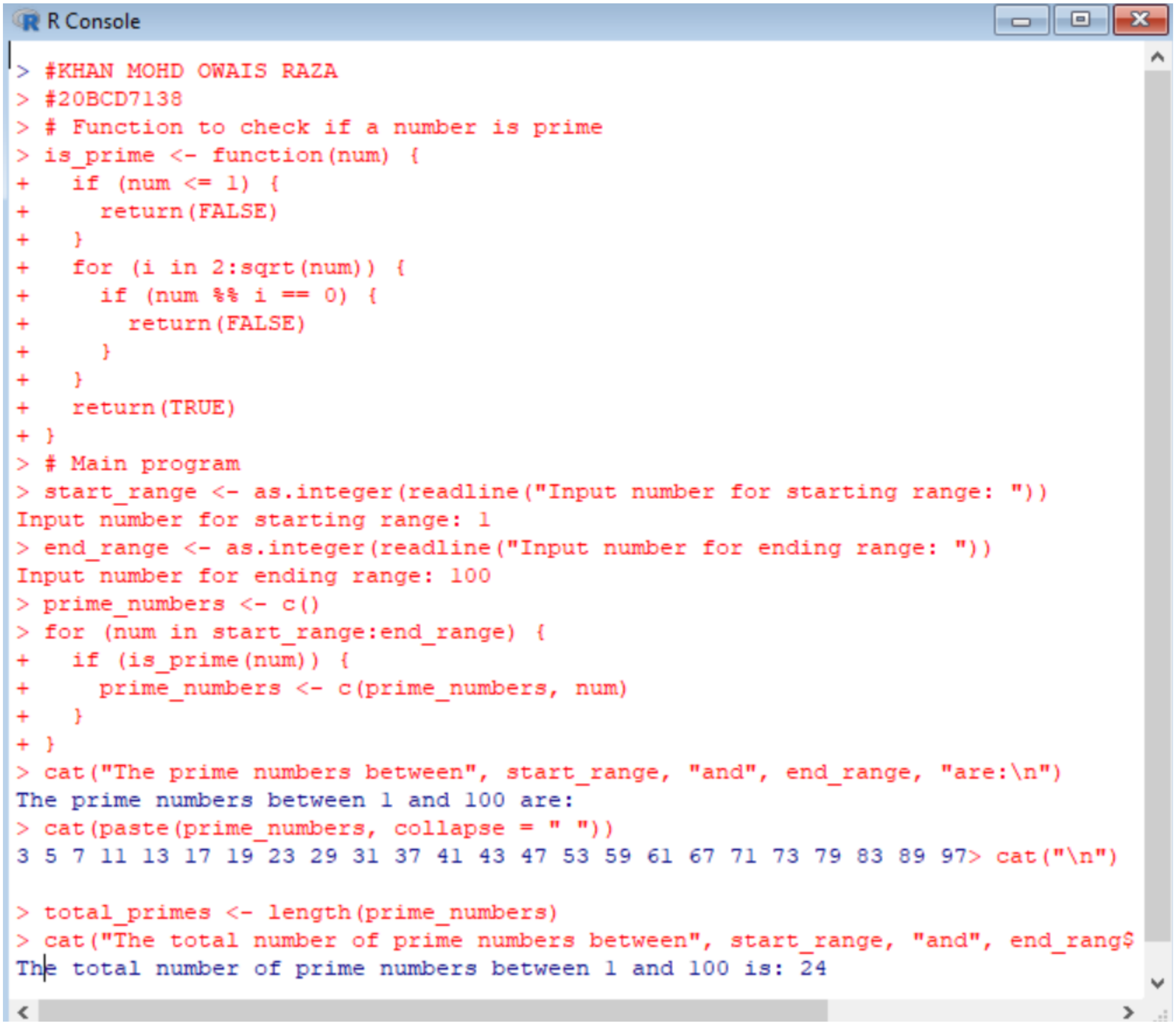
```r
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to check if a number is prime
is_prime <- function(num) {
  if (num <= 1) {
    return(FALSE)
  }
  for (i in 2:sqrt(num)) {
    if (num %% i == 0) {
      return(FALSE)
    }
  }
  return(TRUE)
}
# Main program
num <- as.integer(readline("Input a number to check prime or not:
"))
if (is_prime(num)) {
  print("The entered number is a prime number.")
} else {
  print("The entered number is not a prime number.")
}
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to check if a number is prime
> is_prime <- function(num) {
+   if (num <= 1) {
+     return(FALSE)
+   }
+   for (i in 2:sqrt(num)) {
+     if (num %% i == 0) {
+       return(FALSE)
+     }
+   }
+   return(TRUE)
+ }
> # Main program
> num <- as.integer(readline("Input a number to check prime or not: "))
Input a number to check prime or not: 13
> if (is_prime(num)) {
+   print("The entered number is a prime number.")
+ } else {
+   print("The entered number is not a prime number.")
+ }
[1] "The entered number is a prime number."
> 
```

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

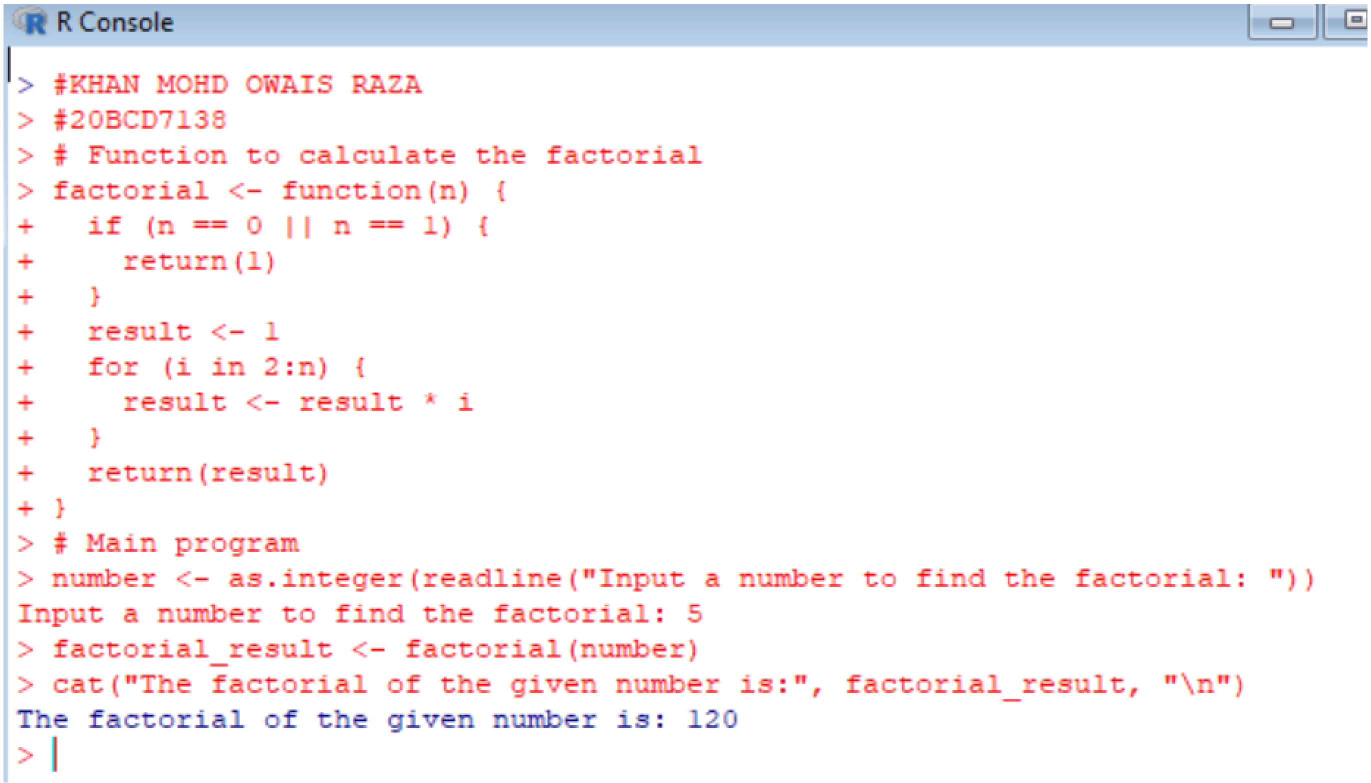
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to check if a number is prime
is_prime <- function(num) {
  if (num <= 1) {
    return(FALSE)
  }
  for (i in 2:sqrt(num)) {
    if (num %% i == 0) {
      return(FALSE)
    }
  }
  return(TRUE)
}
# Main program
start_range <- as.integer(readline("Input number for starting
range: "))
end_range <- as.integer(readline("Input number for ending range:
"))
prime_numbers <- c()
for (num in start_range:end_range) {
  if (is_prime(num)) {
    prime_numbers <- c(prime_numbers, num)
  }
}
cat("The prime numbers between", start_range, "and", end_range,
"are:\n")
cat(paste(prime_numbers, collapse = " "))
cat("\n")
total_primes <- length(prime_numbers)
cat("The total number of prime numbers between", start_range,
"and", end_range, "is:", total_primes, "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to check if a number is prime
> is_prime <- function(num) {
+   if (num <= 1) {
+     return(FALSE)
+   }
+   for (i in 2:sqrt(num)) {
+     if (num %% i == 0) {
+       return(FALSE)
+     }
+   }
+   return(TRUE)
+ }
> # Main program
> start_range <- as.integer(readline("Input number for starting range: "))
Input number for starting range: 1
> end_range <- as.integer(readline("Input number for ending range: "))
Input number for ending range: 100
> prime_numbers <- c()
> for (num in start_range:end_range) {
+   if (is_prime(num)) {
+     prime_numbers <- c(prime_numbers, num)
+   }
+ }
> cat("The prime numbers between", start_range, "and", end_range, "are:\n")
The prime numbers between 1 and 100 are:
> cat(paste(prime_numbers, collapse = " "))
3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97> cat("\n")

> total_primes <- length(prime_numbers)
> cat("The total number of prime numbers between", start_range, "and", end_rang$
The total number of prime numbers between 1 and 100 is: 24
```

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

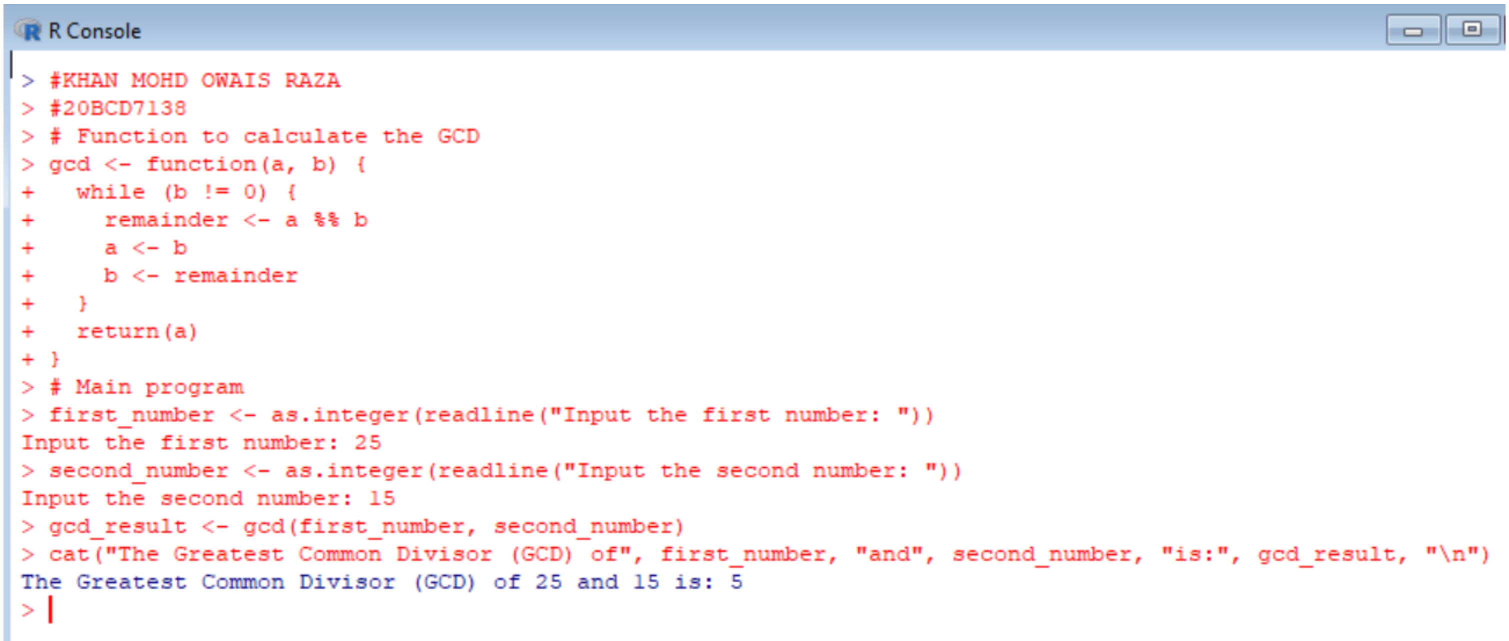
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the factorial
factorial <- function(n) {
  if (n == 0 || n == 1) {
    return(1)
  }
  result <- 1
  for (i in 2:n) {
    result <- result * i
  }
  return(result)
}
# Main program
number <- as.integer(readline("Input a number to find the
factorial: "))
factorial_result <- factorial(number)
cat("The factorial of the given number is:", factorial_result,
"\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the factorial
> factorial <- function(n) {
+   if (n == 0 || n == 1) {
+     return(1)
+   }
+   result <- 1
+   for (i in 2:n) {
+     result <- result * i
+   }
+   return(result)
+ }
> # Main program
> number <- as.integer(readline("Input a number to find the factorial: "))
Input a number to find the factorial: 5
> factorial_result <- factorial(number)
> cat("The factorial of the given number is:", factorial_result, "\n")
The factorial of the given number is: 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

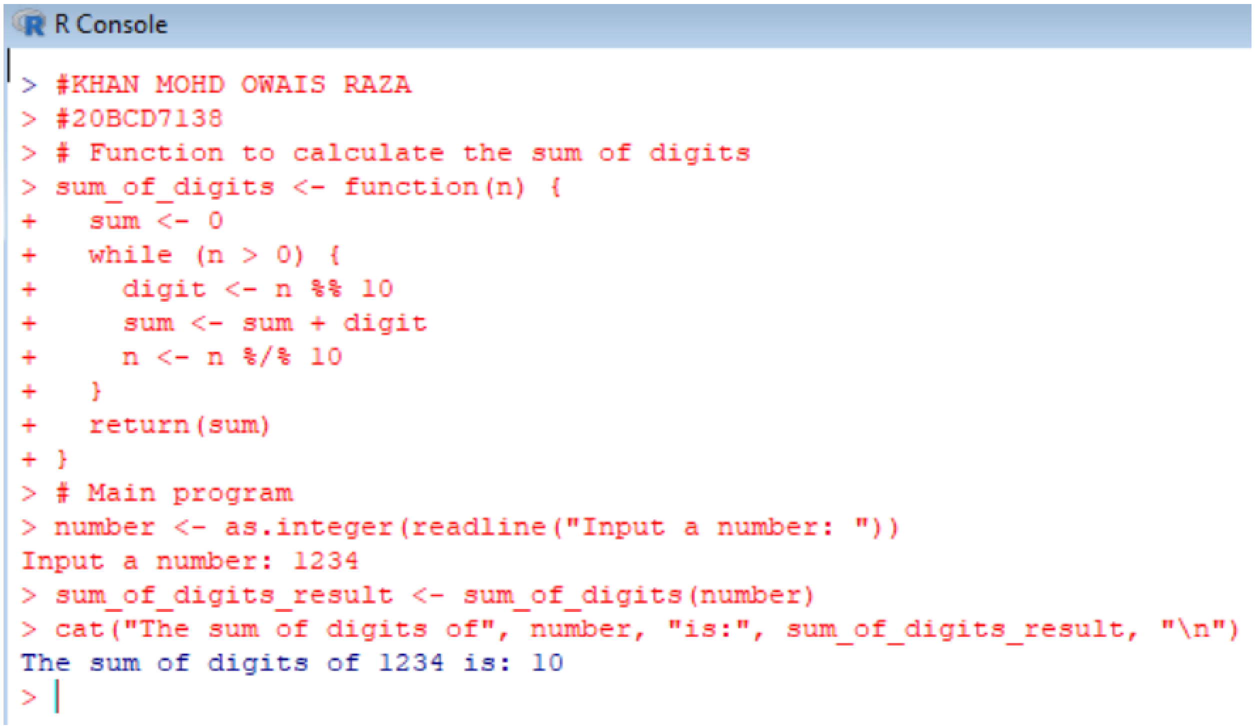
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the GCD
gcd <- function(a, b) {
  while (b != 0) {
    remainder <- a %% b
    a <- b
    b <- remainder
  }
  return(a)
}
# Main program
first_number <- as.integer(readline("Input the first number: "))
second_number <- as.integer(readline("Input the second number:
"))
gcd_result <- gcd(first_number, second_number)
cat("The Greatest Common Divisor (GCD) of", first_number, "and",
second_number, "is:", gcd_result, "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the GCD
> gcd <- function(a, b) {
+   while (b != 0) {
+     remainder <- a %% b
+     a <- b
+     b <- remainder
+   }
+   return(a)
+ }
> # Main program
> first_number <- as.integer(readline("Input the first number: "))
Input the first number: 25
> second_number <- as.integer(readline("Input the second number: "))
Input the second number: 15
> gcd_result <- gcd(first_number, second_number)
> cat("The Greatest Common Divisor (GCD) of", first_number, "and", second_number, "is:", gcd_result, "\n")
The Greatest Common Divisor (GCD) of 25 and 15 is: 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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the sum of digits
sum_of_digits <- function(n) {
  sum <- 0
  while (n > 0) {
    digit <- n %% 10
    sum <- sum + digit
    n <- n %/% 10
  }
  return(sum)
}
# Main program
number <- as.integer(readline("Input a number: "))
sum_of_digits_result <- sum_of_digits(number)
cat("The sum of digits of", number, "is:", sum_of_digits_result,
"\n")
```

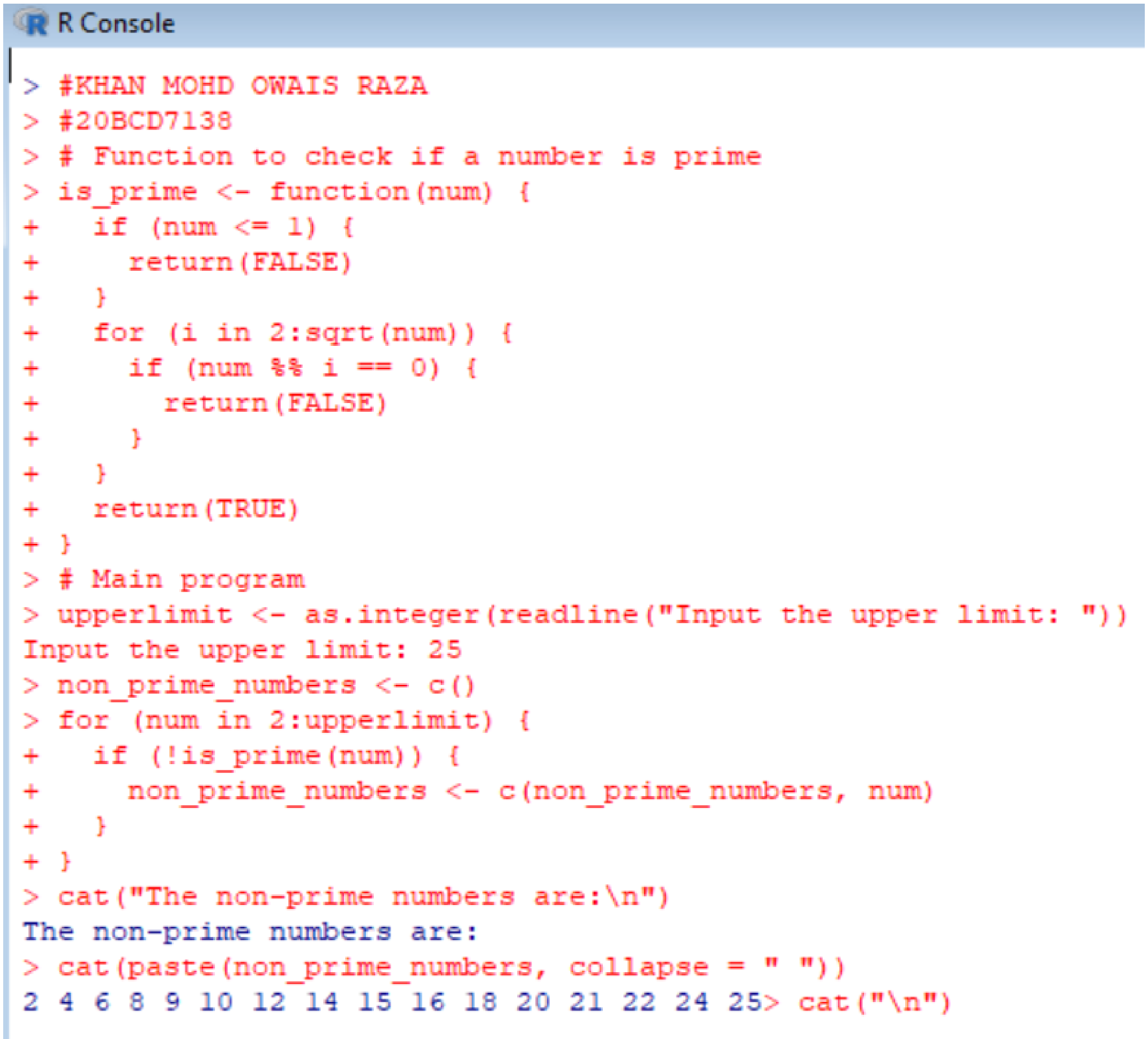


R Console

```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the sum of digits
> sum_of_digits <- function(n) {
+   sum <- 0
+   while (n > 0) {
+     digit <- n %% 10
+     sum <- sum + digit
+     n <- n %/% 10
+   }
+   return(sum)
+ }
> # Main program
> number <- as.integer(readline("Input a number: "))
Input a number: 1234
> sum_of_digits_result <- sum_of_digits(number)
> cat("The sum of digits of", number, "is:", sum_of_digits_result, "\n")
The sum of digits of 1234 is: 10
> 
```

7] Write a program in R to list non-prime numbers from 1 to an upper bound.
Sample Output:
Input the upper limit: 25

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to check if a number is prime
is_prime <- function(num) {
  if (num <= 1) {
    return(FALSE)
  }
  for (i in 2:sqrt(num)) {
    if (num %% i == 0) {
      return(FALSE)
    }
  }
  return(TRUE)
}
# Main program
upperlimit <- as.integer(readline("Input the upper limit: "))
non_prime_numbers <- c()
for (num in 2:upperlimit) {
  if (!is_prime(num)) {
    non_prime_numbers <- c(non_prime_numbers, num)
  }
}
cat("The non-prime numbers are:\n")
cat(paste(non_prime_numbers, collapse = " "))
cat("\n")
```



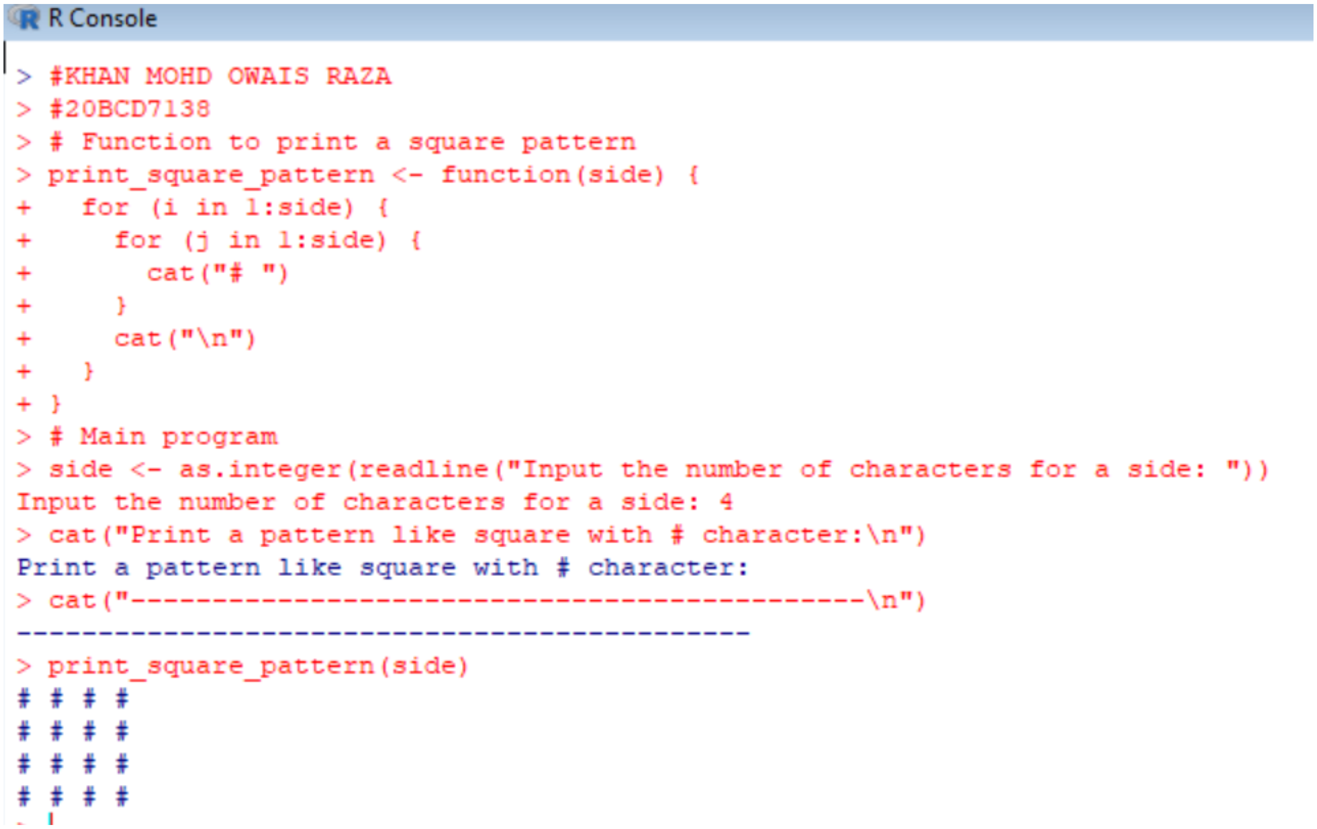
```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to check if a number is prime
> is_prime <- function(num) {
+   if (num <= 1) {
+     return(FALSE)
+   }
+   for (i in 2:sqrt(num)) {
+     if (num %% i == 0) {
+       return(FALSE)
+     }
+   }
+   return(TRUE)
+ }
> # Main program
> upperlimit <- as.integer(readline("Input the upper limit: "))
Input the upper limit: 25
> non_prime_numbers <- c()
> for (num in 2:upperlimit) {
+   if (!is_prime(num)) {
+     non_prime_numbers <- c(non_prime_numbers, num)
+   }
+ }
> cat("The non-prime numbers are:\n")
The non-prime numbers are:
> cat(paste(non_prime_numbers, collapse = " "))
2 4 6 8 9 10 12 14 15 16 18 20 21 22 24 25> cat("\n")
```

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
####
####
####
####
```

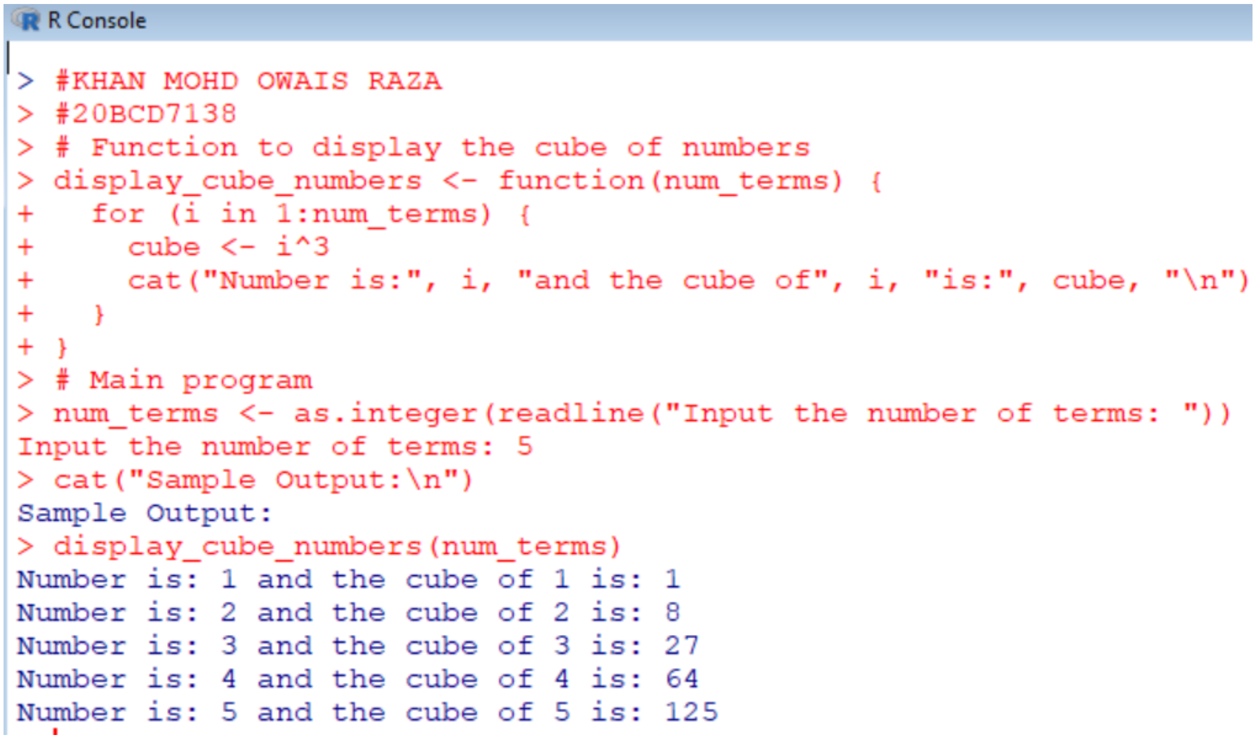
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to print a square pattern
print_square_pattern <- function(side) {
  for (i in 1:side) {
    for (j in 1:side) {
      cat("# ")
    }
    cat("\n")
  }
}
# Main program
side <- as.integer(readline("Input the number of characters for a
side: "))
cat("Print a pattern like square with # character:\n")
cat("---------------------------------------------\n")
print_square_pattern(side)
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to print a square pattern
> print_square_pattern <- function(side) {
+   for (i in 1:side) {
+     for (j in 1:side) {
+       cat("# ")
+     }
+     cat("\n")
+   }
+ }
> # Main program
> side <- as.integer(readline("Input the number of characters for a side: "))
Input the number of characters for a side: 4
> cat("Print a pattern like square with # character:\n")
Print a pattern like square with # character:
> cat("---------------------------------------------\n")
---------------------------------------------
> print_square_pattern(side)
# # # #
# # # #
# # # #
# # # #
```

9] Write a program in R to display the cube of the number upto given 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

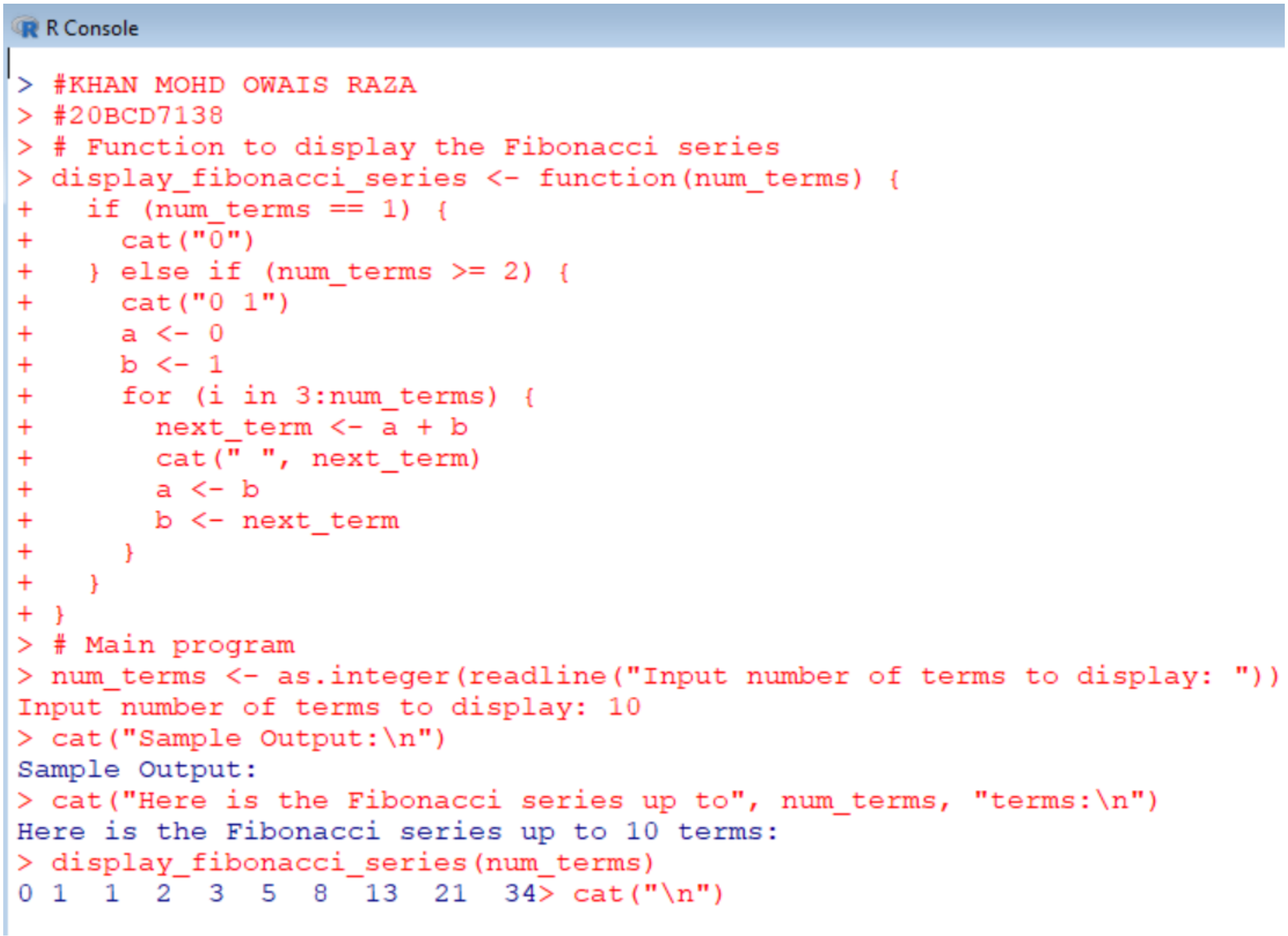
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the cube of numbers
display_cube_numbers <- function(num_terms) {
  for (i in 1:num_terms) {
    cube <- i^3
    cat("Number is:", i, "and the cube of", i, "is:", cube, "\n")
  }
}
# Main program
num_terms <- as.integer(readline("Input the number of terms: "))
cat("Sample Output:\n")
display_cube_numbers(num_terms)
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the cube of numbers
> display_cube_numbers <- function(num_terms) {
+   for (i in 1:num_terms) {
+     cube <- i^3
+     cat("Number is:", i, "and the cube of", i, "is:", cube, "\n")
+   }
+ }
> # Main program
> num_terms <- as.integer(readline("Input the number of terms: "))
Input the number of terms: 5
> cat("Sample Output:\n")
Sample Output:
> display_cube_numbers(num_terms)
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
```

10] Write a program in R to display the first n terms of the 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

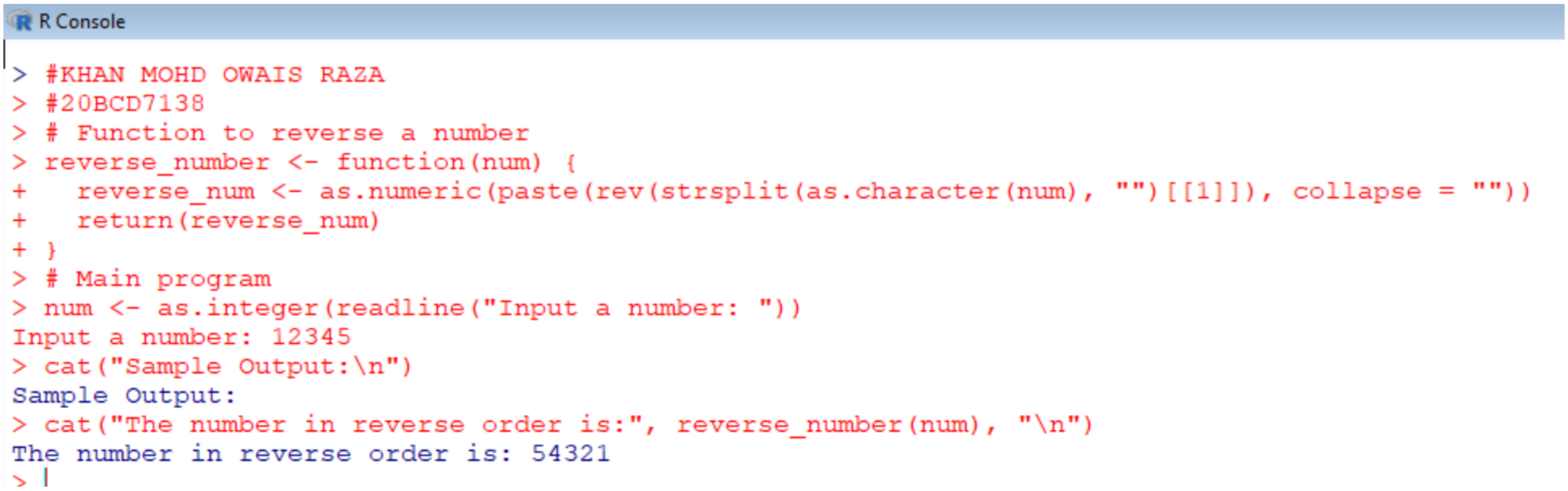
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the Fibonacci series
display_fibonacci_series <- function(num_terms) {
  if (num_terms == 1) {
    cat("0")
  } else if (num_terms >= 2) {
    cat("0 1")
    a <- 0
    b <- 1
    for (i in 3:num_terms) {
      next_term <- a + b
      cat(" ", next_term)
      a <- b
      b <- next_term
    }
  }
}
# Main program
num_terms <- as.integer(readline("Input number of terms to
display: "))
cat("Sample Output:\n")
cat("Here is the Fibonacci series up to", num_terms, "terms:\n")
display_fibonacci_series(num_terms)
cat("\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the Fibonacci series
> display_fibonacci_series <- function(num_terms) {
+   if (num_terms == 1) {
+     cat("0")
+   } else if (num_terms >= 2) {
+     cat("0 1")
+     a <- 0
+     b <- 1
+     for (i in 3:num_terms) {
+       next_term <- a + b
+       cat(" ", next_term)
+       a <- b
+       b <- next_term
+     }
+   }
+ }
> # Main program
> num_terms <- as.integer(readline("Input number of terms to display: "))
Input number of terms to display: 10
> cat("Sample Output:\n")
Sample Output:
> cat("Here is the Fibonacci series up to", num_terms, "terms:\n")
Here is the Fibonacci series up to 10 terms:
> display_fibonacci_series(num_terms)
0 1  1  2  3  5  8  13  21  34> cat("\n")
```

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

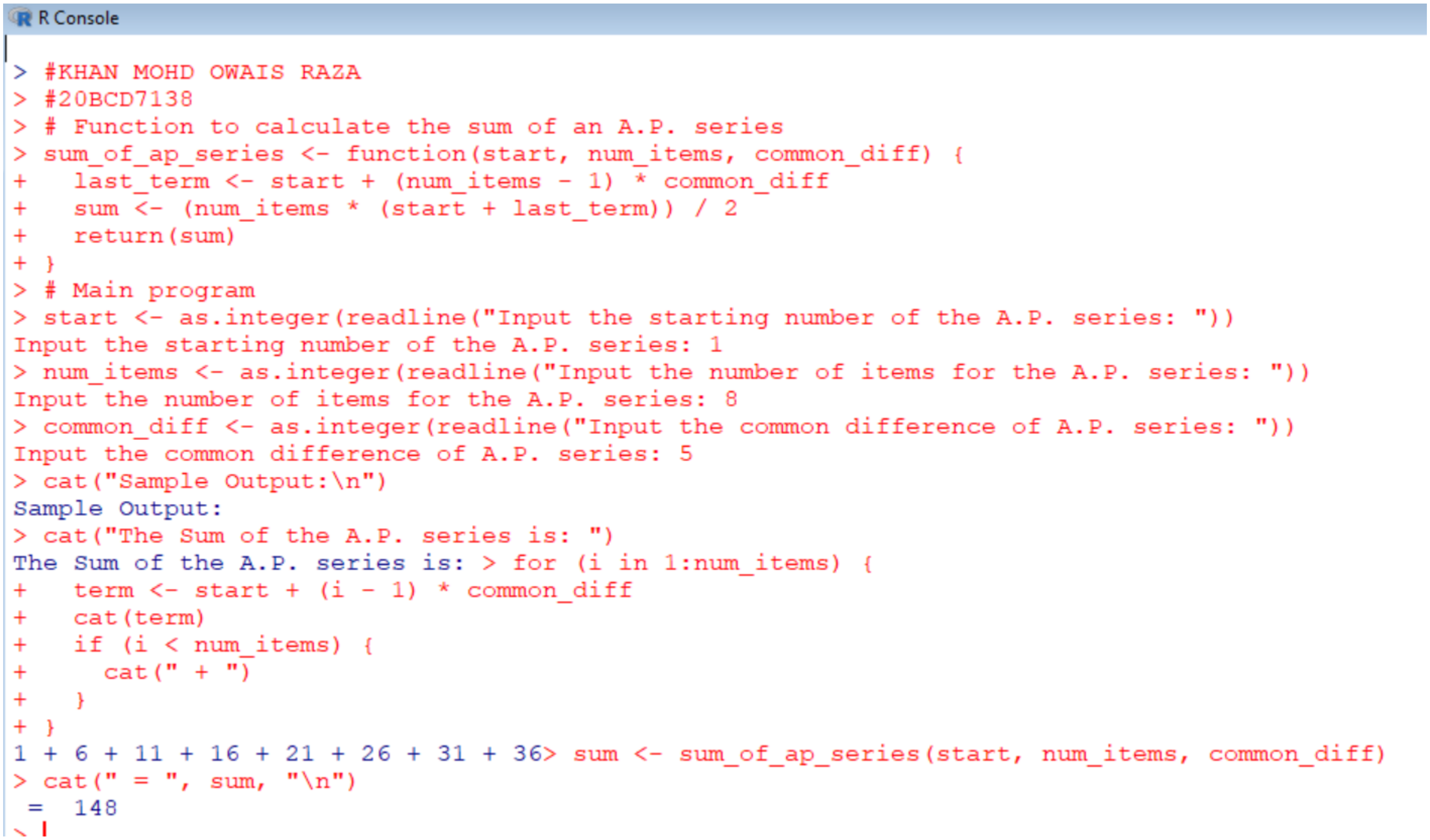
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to reverse a number
reverse_number <- function(num) {
  reverse_num <- as.numeric(paste(rev(strsplit(as.character(num),
"")[[1]]), collapse = ""))
  return(reverse_num)
}
# Main program
num <- as.integer(readline("Input a number: "))
cat("Sample Output:\n")
cat("The number in reverse order is:", reverse_number(num), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to reverse a number
> reverse_number <- function(num) {
+   reverse_num <- as.numeric(paste(rev(strsplit(as.character(num), "")[[1]]), collapse = ""))
+   return(reverse_num)
+ }
> # Main program
> num <- as.integer(readline("Input a number: "))
Input a number: 12345
> cat("Sample Output:\n")
Sample Output:
> cat("The number in reverse order is:", reverse_number(num), "\n")
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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the sum of an A.P. series
sum_of_ap_series <- function(start, num_items, common_diff) {
  last_term <- start + (num_items - 1) * common_diff
  sum <- (num_items * (start + last_term)) / 2
  return(sum)
}
# Main program
start <- as.integer(readline("Input the starting number of the
A.P. series: "))
num_items <- as.integer(readline("Input the number of items for
the A.P. series: "))
common_diff <- as.integer(readline("Input the common difference
of A.P. series: "))
cat("Sample Output:\n")
cat("The Sum of the A.P. series is: ")
for (i in 1:num_items) {
  term <- start + (i - 1) * common_diff
  cat(term)
  if (i < num_items) {
    cat(" + ")
  }
}
sum <- sum_of_ap_series(start, num_items, common_diff)
cat(" = ", sum, "\n")
```

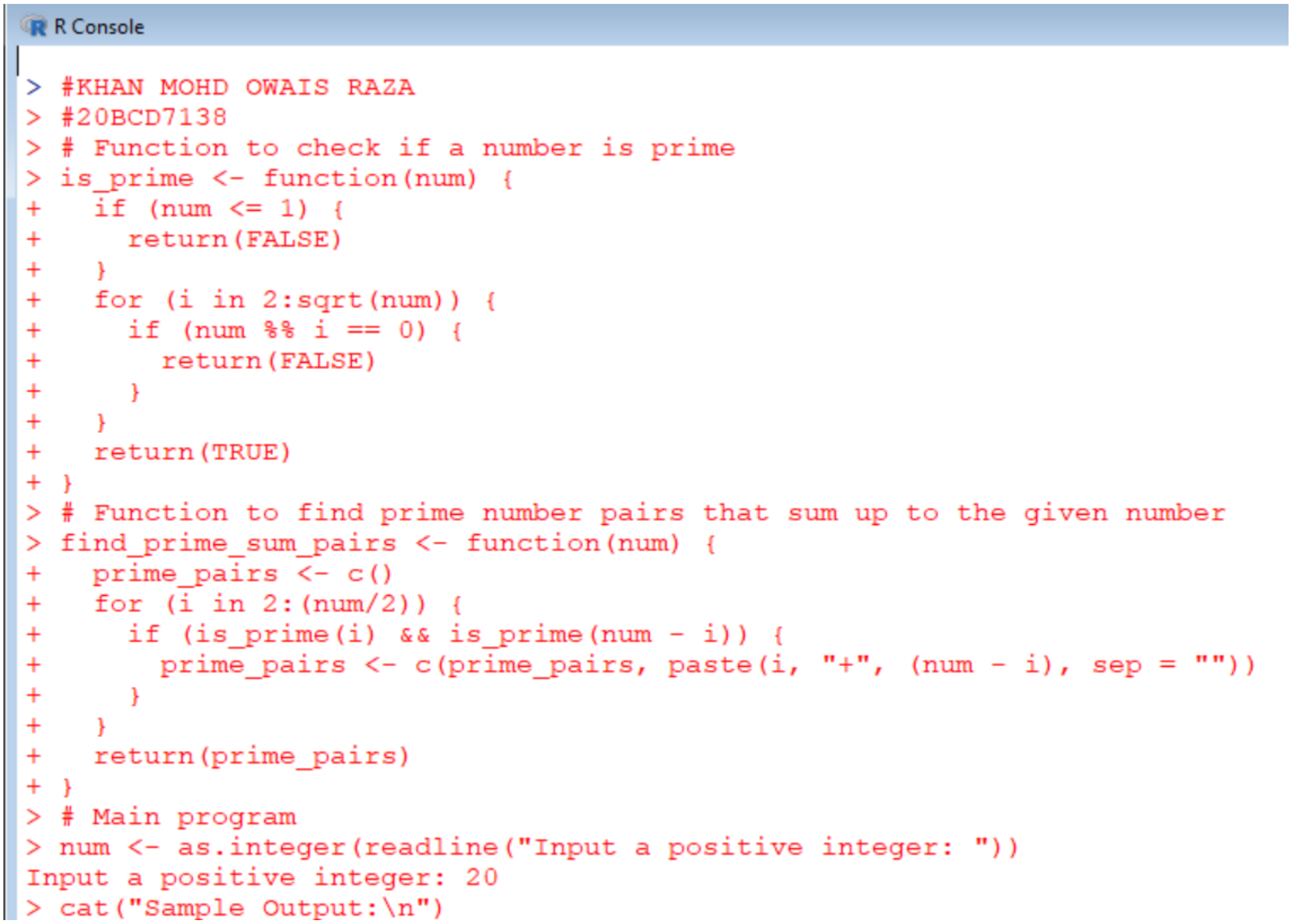


```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the sum of an A.P. series
> sum_of_ap_series <- function(start, num_items, common_diff) {
+   last_term <- start + (num_items - 1) * common_diff
+   sum <- (num_items * (start + last_term)) / 2
+   return(sum)
+ }
> # Main program
> start <- as.integer(readline("Input the starting number of the A.P. series: "))
Input the starting number of the A.P. series: 1
> num_items <- as.integer(readline("Input the number of items for the A.P. series: "))
Input the number of items for the A.P. series: 8
> common_diff <- as.integer(readline("Input the common difference of A.P. series: "))
Input the common difference of A.P. series: 5
> cat("Sample Output:\n")
Sample Output:
> cat("The Sum of the A.P. series is: ")
The Sum of the A.P. series is: > for (i in 1:num_items) {
+   term <- start + (i - 1) * common_diff
+   cat(term)
+   if (i < num_items) {
+     cat(" + ")
+   }
+ }
1 + 6 + 11 + 16 + 21 + 26 + 31 + 36> sum <- sum_of_ap_series(start, num_items, common_diff)
> cat(" = ", sum, "\n")
 =  148
> 
```

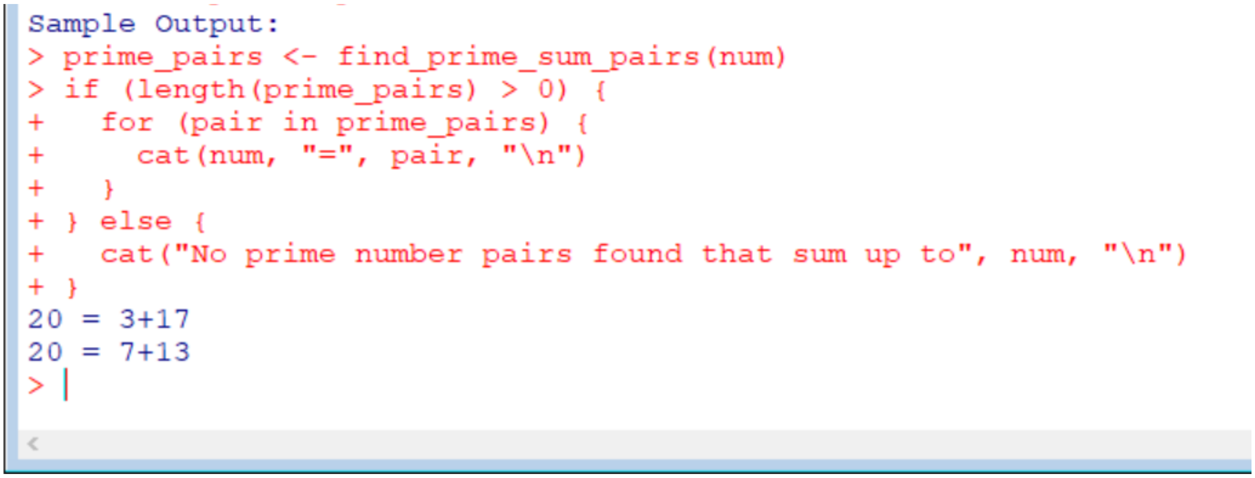
13] Write a program in R to check whether a number can be expressed as the sum of two Prime Numbers.
Sample Output:
Input a positive integer: 20
20 = 3 + 17
20 = 7 + 13

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to check if a number is prime
is_prime <- function(num) {
  if (num <= 1) {
    return(FALSE)
  }
  for (i in 2:sqrt(num)) {
    if (num %% i == 0) {
      return(FALSE)
    }
  }
  return(TRUE)
}
# Function to find prime number pairs that sum up to the given
number
```

```
find_prime_sum_pairs <- function(num) {
  prime_pairs <- c()
  for (i in 2:(num/2)) {
    if (is_prime(i) && is_prime(num - i)) {
      prime_pairs <- c(prime_pairs, paste(i, "+", (num - i), sep
= ""))
    }
  }
  return(prime_pairs)
}
# Main program
num <- as.integer(readline("Input a positive integer: "))
cat("Sample Output:\n")
prime_pairs <- find_prime_sum_pairs(num)
if (length(prime_pairs) > 0) {
  for (pair in prime_pairs) {
    cat(num, "=", pair, "\n")
  }
} else {
  cat("No prime number pairs found that sum up to", num, "\n")
}
```



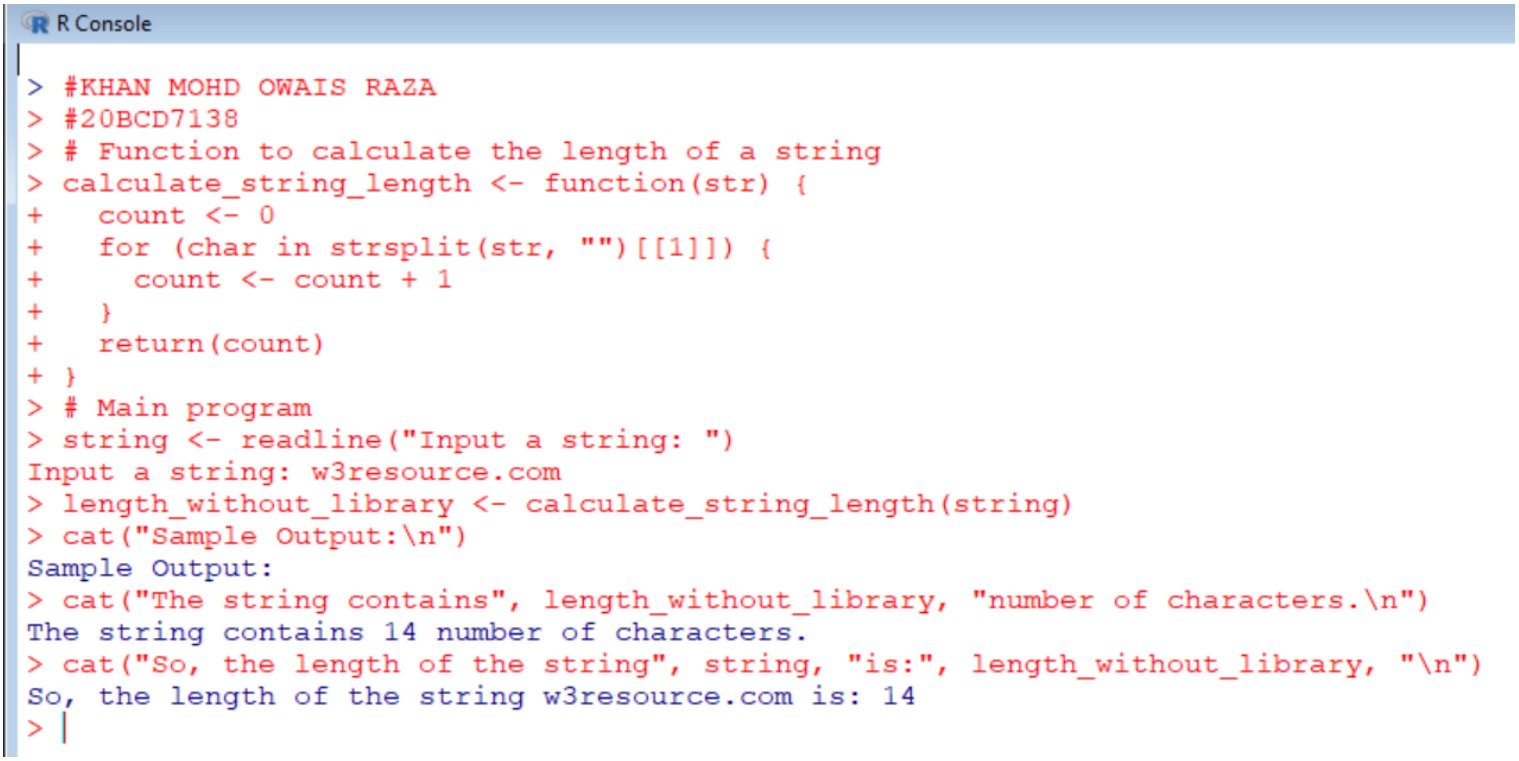
```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to check if a number is prime
> is_prime <- function(num) {
+   if (num <= 1) {
+     return(FALSE)
+   }
+   for (i in 2:sqrt(num)) {
+     if (num %% i == 0) {
+       return(FALSE)
+     }
+   }
+   return(TRUE)
+ }
> # Function to find prime number pairs that sum up to the given number
> find_prime_sum_pairs <- function(num) {
+   prime_pairs <- c()
+   for (i in 2:(num/2)) {
+     if (is_prime(i) && is_prime(num - i)) {
+       prime_pairs <- c(prime_pairs, paste(i, "+", (num - i), sep = ""))
+     }
+   }
+   return(prime_pairs)
+ }
> # Main program
> num <- as.integer(readline("Input a positive integer: "))
Input a positive integer: 20
> cat("Sample Output:\n")
```



```
Sample Output:
> prime_pairs <- find_prime_sum_pairs(num)
> if (length(prime_pairs) > 0) {
+   for (pair in prime_pairs) {
+     cat(num, "=", pair, "\n")
+   }
+ } else {
+   cat("No prime number pairs found that sum up to", num, "\n")
+ }
20 = 3+17
20 = 7+13
> |
```

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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the length of a string
calculate_string_length <- function(str) {
  count <- 0
  for (char in strsplit(str, "")[[1]]) {
    count <- count + 1
  }
  return(count)
}
# Main program
string <- readline("Input a string: ")
length_without_library <- calculate_string_length(string)
cat("Sample Output:\n")
cat("The string contains", length_without_library, "number of
characters.\n")
cat("So, the length of the string", string, "is:",
length_without_library, "\n")
```



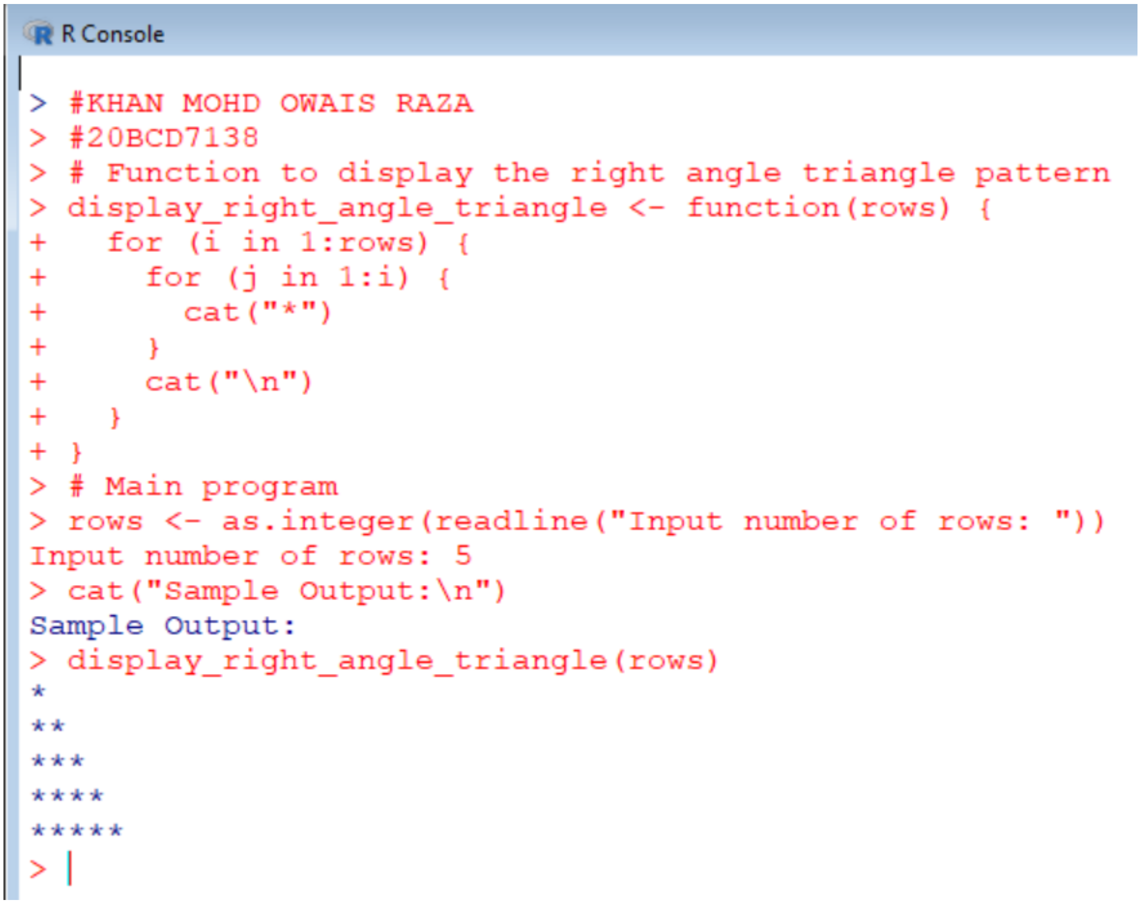
```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the length of a string
> calculate_string_length <- function(str) {
+   count <- 0
+   for (char in strsplit(str, "")[[1]]) {
+     count <- count + 1
+   }
+   return(count)
+ }
> # Main program
> string <- readline("Input a string: ")
Input a string: w3resource.com
> length_without_library <- calculate_string_length(string)
> cat("Sample Output:\n")
Sample Output:
> cat("The string contains", length_without_library, "number of characters.\n")
The string contains 14 number of characters.
> cat("So, the length of the string", string, "is:", length_without_library, "\n")
So, the length of the string w3resource.com is: 14
> 
```

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

Sample Output:

```
Input number of rows: 5
*
**
***
****
*****
```

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the right angle triangle pattern
display_right_angle_triangle <- function(rows) {
  for (i in 1:rows) {
    for (j in 1:i) {
      cat("*")
    }
    cat("\n")
  }
}
# Main program
rows <- as.integer(readline("Input number of rows: "))
cat("Sample Output:\n")
display_right_angle_triangle(rows)
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the right angle triangle pattern
> display_right_angle_triangle <- function(rows) {
+   for (i in 1:rows) {
+     for (j in 1:i) {
+       cat("*")
+     }
+     cat("\n")
+   }
+ }
> # Main program
> rows <- as.integer(readline("Input number of rows: "))
Input number of rows: 5
> cat("Sample Output:\n")
Sample Output:
> display_right_angle_triangle(rows)
*
**
***
****
*****
> 
```

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

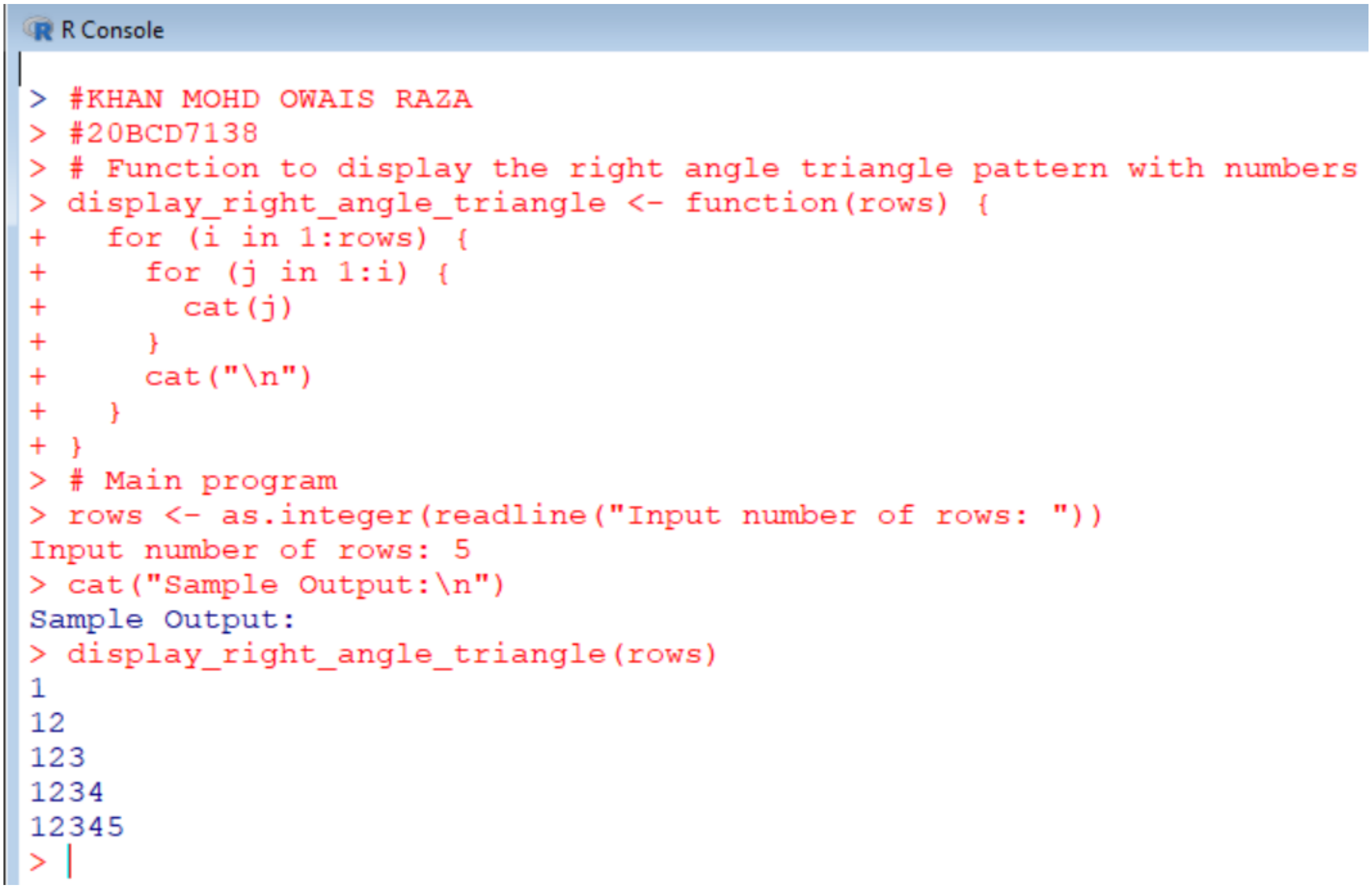
Sample Output:

Input number of rows: 5

```
1
12
123
1234
12345
```

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the right angle triangle pattern with
numbers
display_right_angle_triangle <- function(rows) {
  for (i in 1:rows) {
    for (j in 1:i) {
      cat(j)
    }
    cat("\n")
  }
}
# Main program
rows <- as.integer(readline("Input number of rows: "))
```

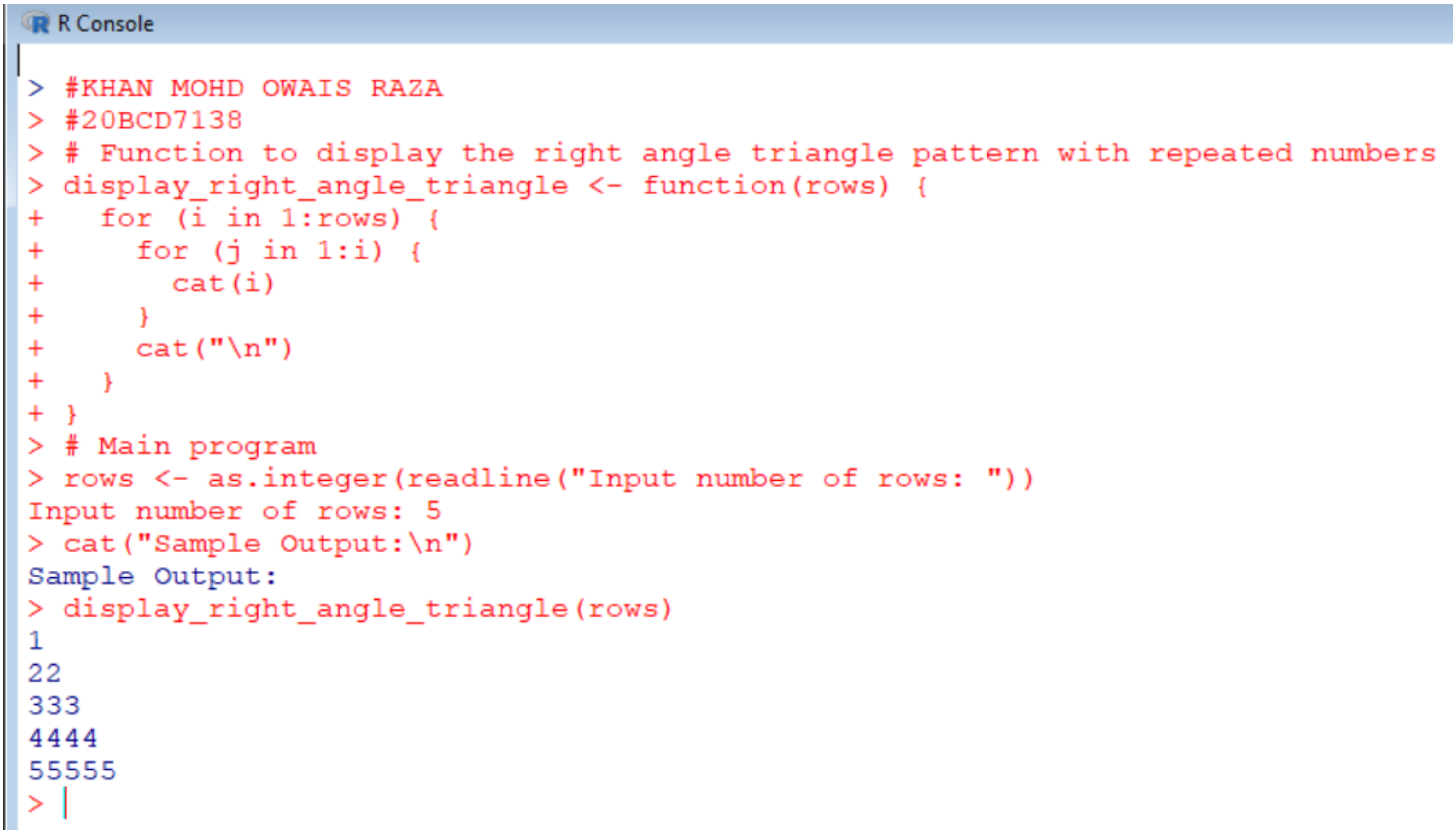
```
cat("Sample Output:\n")
display_right_angle_triangle(rows)
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the right angle triangle pattern with numbers
> display_right_angle_triangle <- function(rows) {
+   for (i in 1:rows) {
+     for (j in 1:i) {
+       cat(j)
+     }
+     cat("\n")
+   }
+ }
> # Main program
> rows <- as.integer(readline("Input number of rows: "))
Input number of rows: 5
> cat("Sample Output:\n")
Sample Output:
> display_right_angle_triangle(rows)
1
12
123
1234
12345
> 
```

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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the right angle triangle pattern with
repeated numbers
display_right_angle_triangle <- function(rows) {
  for (i in 1:rows) {
    for (j in 1:i) {
      cat(i)
    }
    cat("\n")
  }
}
# Main program
rows <- as.integer(readline("Input number of rows: "))
cat("Sample Output:\n")
display_right_angle_triangle(rows)
```

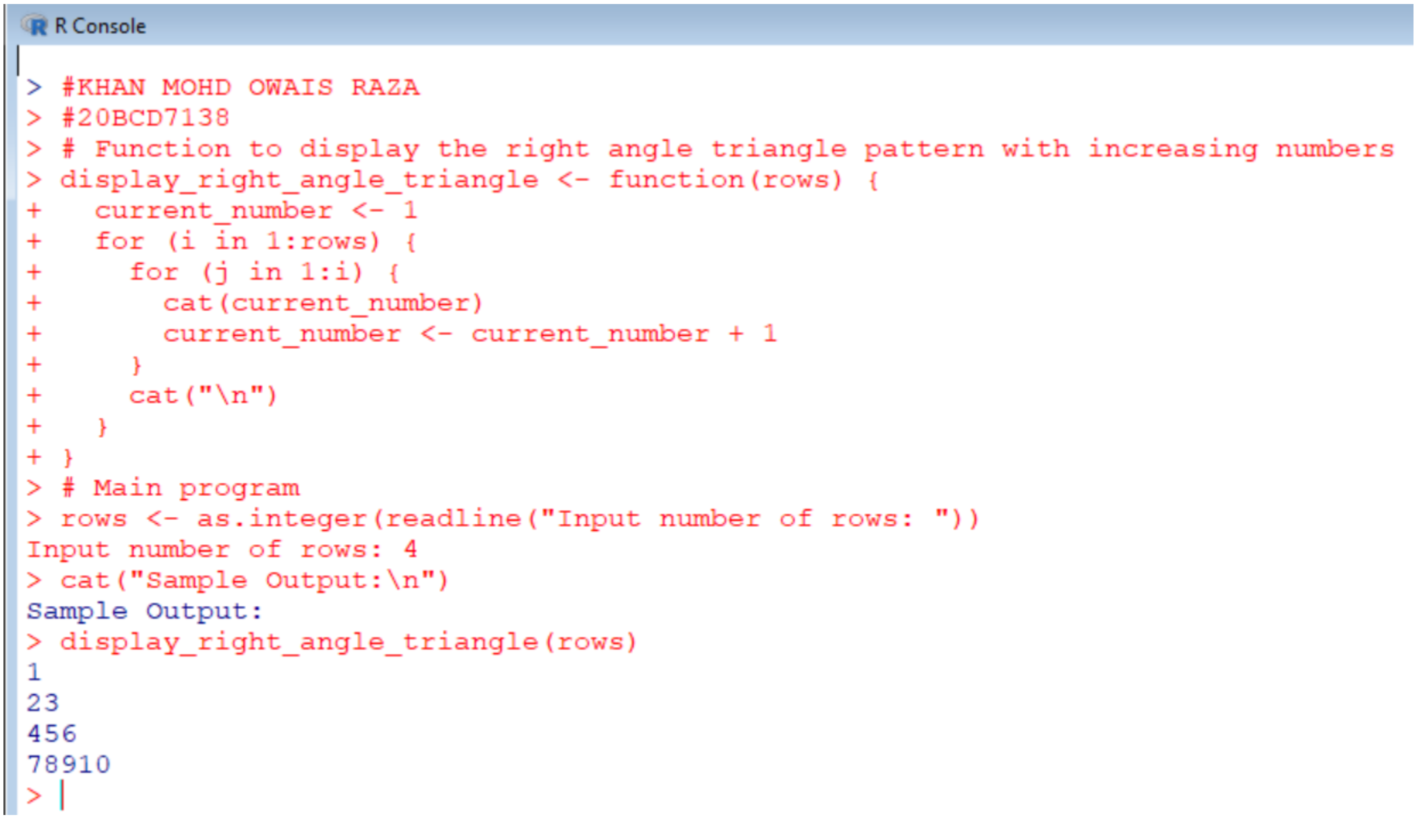


```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the right angle triangle pattern with repeated numbers
> display_right_angle_triangle <- function(rows) {
+   for (i in 1:rows) {
+     for (j in 1:i) {
+       cat(i)
+     }
+     cat("\n")
+   }
+ }
> # Main program
> rows <- as.integer(readline("Input number of rows: "))
Input number of rows: 5
> cat("Sample Output:\n")
Sample Output:
> display_right_angle_triangle(rows)
1
22
333
4444
55555
> 
```

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

Sample Output:
Input number of rows: 4
1
23
456
78910

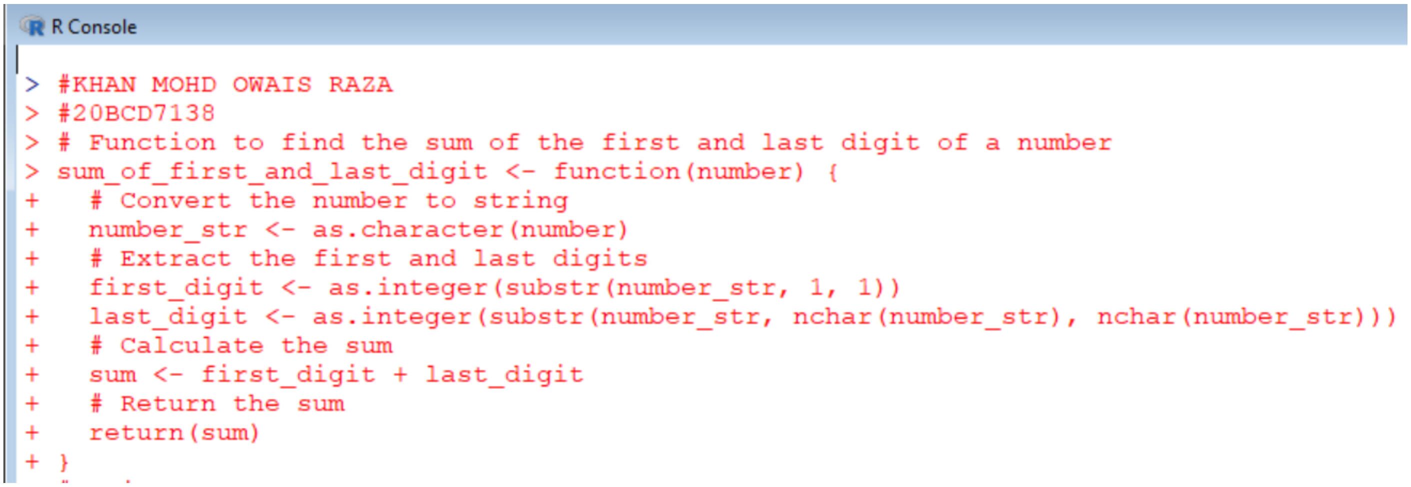
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to display the right angle triangle pattern with
increasing numbers
display_right_angle_triangle <- function(rows) {
  current_number <- 1
  for (i in 1:rows) {
    for (j in 1:i) {
      cat(current_number)
      current_number <- current_number + 1
    }
    cat("\n")
  }
}
# Main program
rows <- as.integer(readline("Input number of rows: "))
cat("Sample Output:\n")
display_right_angle_triangle(rows)
```



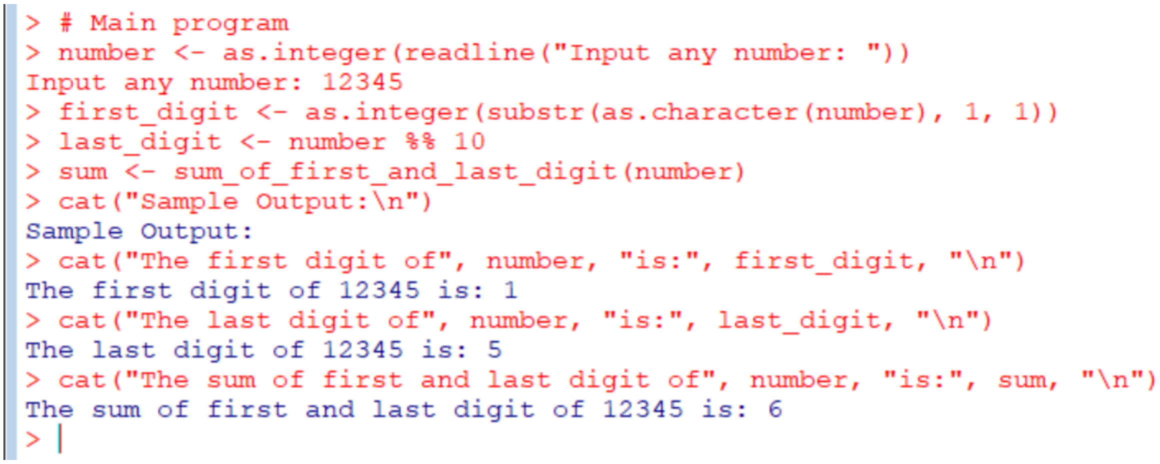
```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to display the right angle triangle pattern with increasing numbers
> display_right_angle_triangle <- function(rows) {
+   current_number <- 1
+   for (i in 1:rows) {
+     for (j in 1:i) {
+       cat(current_number)
+       current_number <- current_number + 1
+     }
+     cat("\n")
+   }
+ }
> # Main program
> rows <- as.integer(readline("Input number of rows: "))
Input number of rows: 4
> cat("Sample Output:\n")
Sample Output:
> display_right_angle_triangle(rows)
1
23
456
78910
> 
```

19] Write a program in R to find the sum of the 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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to find the sum of the first and last digit of a number
sum_of_first_and_last_digit <- function(number) {
  # Convert the number to string
  number_str <- as.character(number)
  # Extract the first and last digits
  first_digit <- as.integer(substr(number_str, 1, 1))
  last_digit <- as.integer(substr(number_str, nchar(number_str), nchar(number_str)))
  # Calculate the sum
  sum <- first_digit + last_digit
  # Return the sum
  return(sum)
}
# Main program
number <- as.integer(readline("Input any number: "))
first_digit <- as.integer(substr(as.character(number), 1, 1))
last_digit <- number %% 10
sum <- sum_of_first_and_last_digit(number)
cat("Sample Output:\n")
cat("The first digit of", number, "is:", first_digit, "\n")
cat("The last digit of", number, "is:", last_digit, "\n")
cat("The sum of first and last digit of", number, "is:", sum, "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to find the sum of the first and last digit of a number
> sum_of_first_and_last_digit <- function(number) {
+   # Convert the number to string
+   number_str <- as.character(number)
+   # Extract the first and last digits
+   first_digit <- as.integer(substr(number_str, 1, 1))
+   last_digit <- as.integer(substr(number_str, nchar(number_str), nchar(number_str)))
+   # Calculate the sum
+   sum <- first_digit + last_digit
+   # Return the sum
+   return(sum)
+ }
```

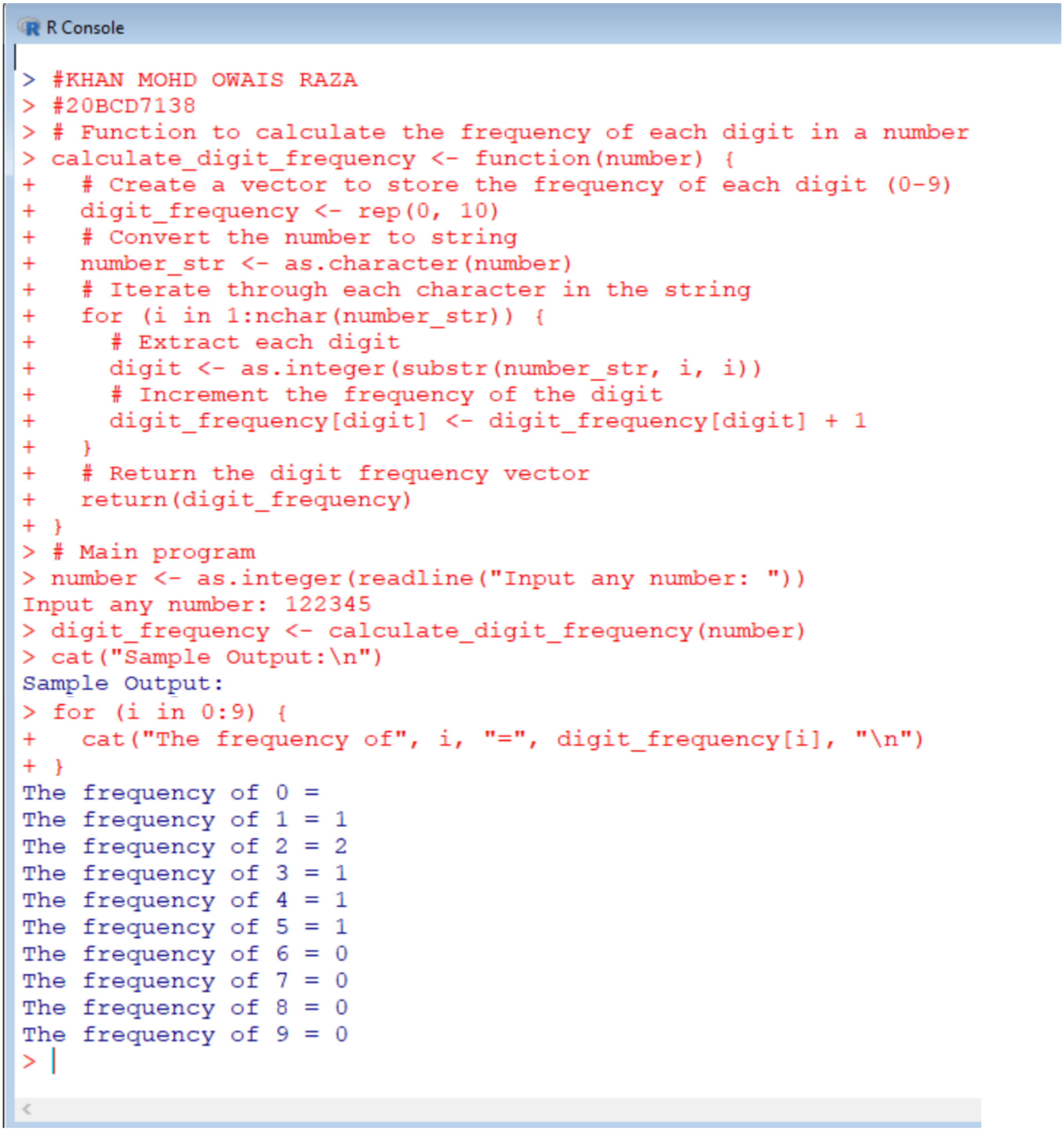


```
> # Main program
> number <- as.integer(readline("Input any number: "))
Input any number: 12345
> first_digit <- as.integer(substr(as.character(number), 1, 1))
> last_digit <- number %% 10
> sum <- sum_of_first_and_last_digit(number)
> cat("Sample Output:\n")
Sample Output:
> cat("The first digit of", number, "is:", first_digit, "\n")
The first digit of 12345 is: 1
> cat("The last digit of", number, "is:", last_digit, "\n")
The last digit of 12345 is: 5
> cat("The sum of first and last digit of", number, "is:", sum, "\n")
The sum of first and last digit of 12345 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

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to calculate the frequency of each digit in a number
calculate_digit_frequency <- function(number) {
  # Create a vector to store the frequency of each digit (0-9)
  digit_frequency <- rep(0, 10)
  # Convert the number to string
  number_str <- as.character(number)
  # Iterate through each character in the string
  for (i in 1:nchar(number_str)) {
    # Extract each digit
    digit <- as.integer(substr(number_str, i, i))
    # Increment the frequency of the digit
    digit_frequency[digit] <- digit_frequency[digit] + 1
  }
  # Return the digit frequency vector
  return(digit_frequency)
}
# Main program
```

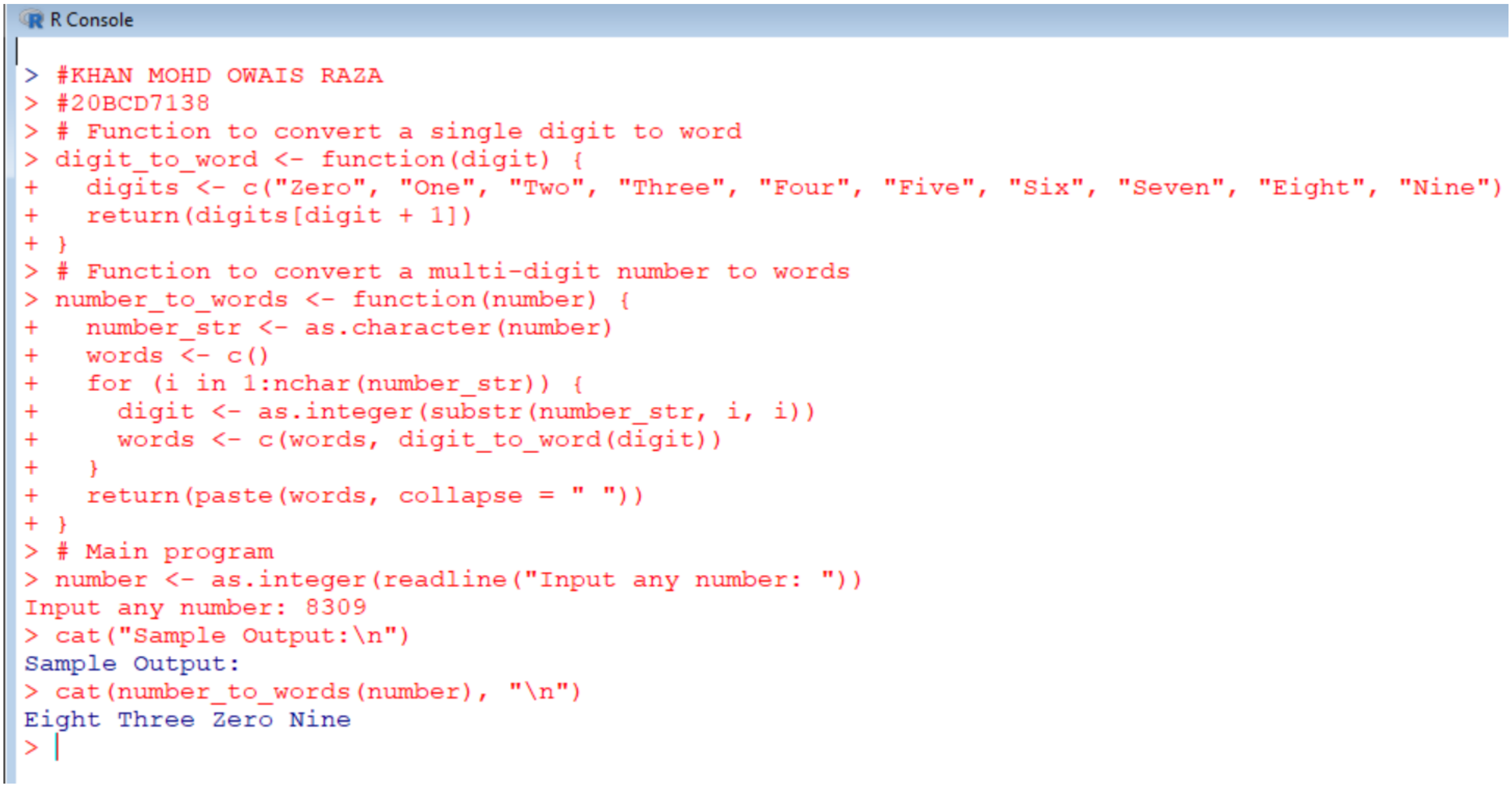
```
number <- as.integer(readline("Input any number: "))
digit_frequency <- calculate_digit_frequency(number)
cat("Sample Output:\n")
for (i in 0:9) {
  cat("The frequency of", i, "=", digit_frequency[i], "\n")
}
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to calculate the frequency of each digit in a number
> calculate_digit_frequency <- function(number) {
+   # Create a vector to store the frequency of each digit (0-9)
+   digit_frequency <- rep(0, 10)
+   # Convert the number to string
+   number_str <- as.character(number)
+   # Iterate through each character in the string
+   for (i in 1:nchar(number_str)) {
+     # Extract each digit
+     digit <- as.integer(substr(number_str, i, i))
+     # Increment the frequency of the digit
+     digit_frequency[digit] <- digit_frequency[digit] + 1
+   }
+   # Return the digit frequency vector
+   return(digit_frequency)
+ }
> # Main program
> number <- as.integer(readline("Input any number: "))
Input any number: 122345
> digit_frequency <- calculate_digit_frequency(number)
> cat("Sample Output:\n")
Sample Output:
> for (i in 0:9) {
+   cat("The frequency of", i, "=", digit_frequency[i], "\n")
+ }
The frequency of 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
> 
```

21] Write a program in R to display the given number in words.
Sample Output:
Input any number: 8309
Eight Three Zero Nine

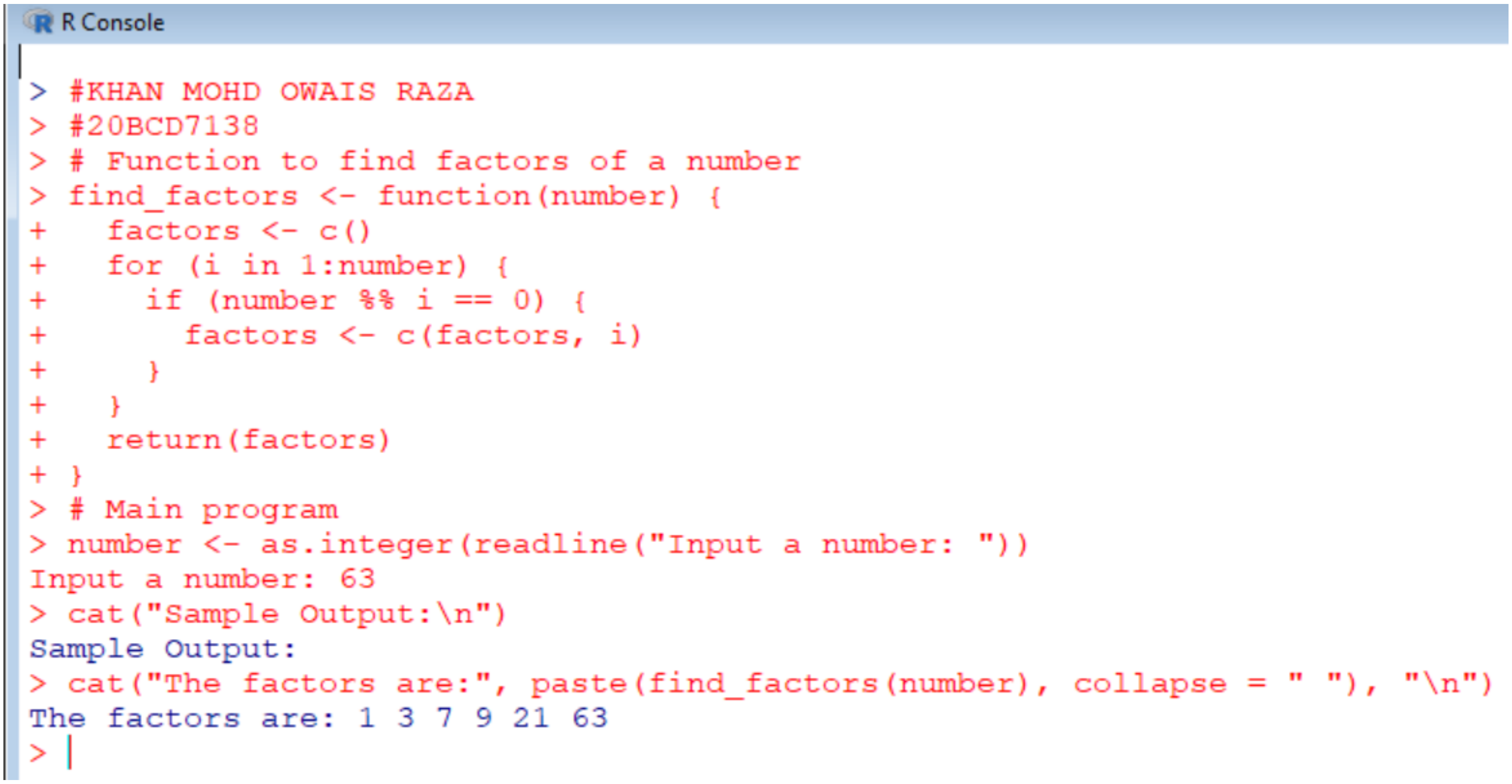
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to convert a single digit to word
digit_to_word <- function(digit) {
  digits <- c("Zero", "One", "Two", "Three", "Four", "Five",
"Six", "Seven", "Eight", "Nine")
  return(digits[digit + 1])
}
# Function to convert a multi-digit number to words
number_to_words <- function(number) {
  number_str <- as.character(number)
  words <- c()
  for (i in 1:nchar(number_str)) {
    digit <- as.integer(substr(number_str, i, i))
    words <- c(words, digit_to_word(digit))
  }
  return(paste(words, collapse = " "))
}
# Main program
number <- as.integer(readline("Input any number: "))
cat("Sample Output:\n")
cat(number_to_words(number), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to convert a single digit to word
> digit_to_word <- function(digit) {
+   digits <- c("Zero", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine")
+   return(digits[digit + 1])
+ }
> # Function to convert a multi-digit number to words
> number_to_words <- function(number) {
+   number_str <- as.character(number)
+   words <- c()
+   for (i in 1:nchar(number_str)) {
+     digit <- as.integer(substr(number_str, i, i))
+     words <- c(words, digit_to_word(digit))
+   }
+   return(paste(words, collapse = " "))
+ }
> # Main program
> number <- as.integer(readline("Input any number: "))
Input any number: 8309
> cat("Sample Output:\n")
Sample Output:
> cat(number_to_words(number), "\n")
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

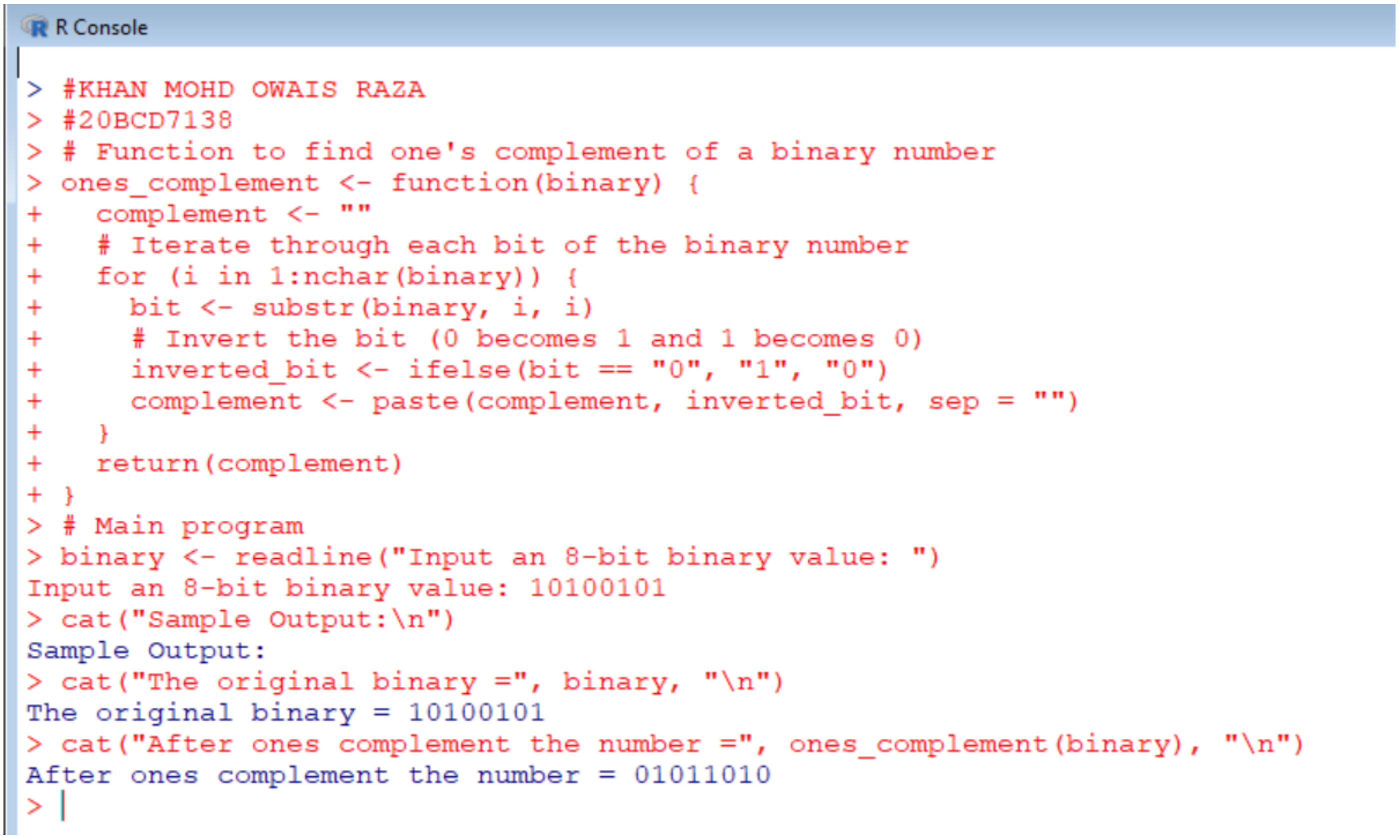
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to find factors of a number
find_factors <- function(number) {
  factors <- c()
  for (i in 1:number) {
    if (number %% i == 0) {
      factors <- c(factors, i)
    }
  }
  return(factors)
}
# Main program
number <- as.integer(readline("Input a number: "))
cat("Sample Output:\n")
cat("The factors are:", paste(find_factors(number), collapse = "
"), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to find factors of a number
> find_factors <- function(number) {
+   factors <- c()
+   for (i in 1:number) {
+     if (number %% i == 0) {
+       factors <- c(factors, i)
+     }
+   }
+   return(factors)
+ }
> # Main program
> number <- as.integer(readline("Input a number: "))
Input a number: 63
> cat("Sample Output:\n")
Sample Output:
> cat("The factors are:", paste(find_factors(number), collapse = " "), "\n")
The factors are: 1 3 7 9 21 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

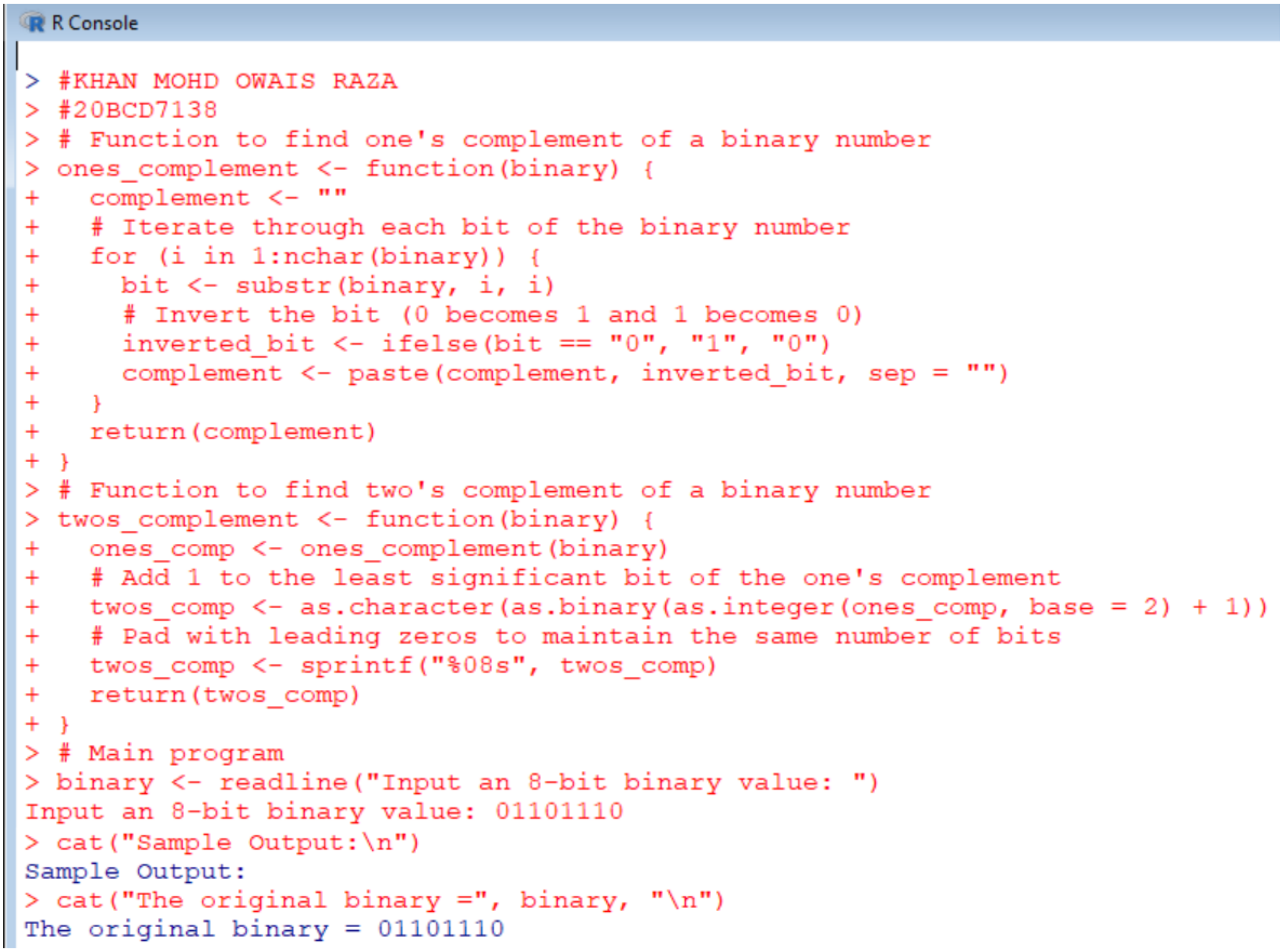
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to find one's complement of a binary number
ones_complement <- function(binary) {
  complement <- ""
  # Iterate through each bit of the binary number
  for (i in 1:nchar(binary)) {
    bit <- substr(binary, i, i)
    # Invert the bit (0 becomes 1 and 1 becomes 0)
    inverted_bit <- ifelse(bit == "0", "1", "0")
    complement <- paste(complement, inverted_bit, sep = "")
  }
  return(complement)
}
# Main program
binary <- readline("Input an 8-bit binary value: ")
cat("Sample Output:\n")
cat("The original binary =", binary, "\n")
cat("After ones complement the number =",
ones_complement(binary), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to find one's complement of a binary number
> ones_complement <- function(binary) {
+   complement <- ""
+   # Iterate through each bit of the binary number
+   for (i in 1:nchar(binary)) {
+     bit <- substr(binary, i, i)
+     # Invert the bit (0 becomes 1 and 1 becomes 0)
+     inverted_bit <- ifelse(bit == "0", "1", "0")
+     complement <- paste(complement, inverted_bit, sep = "")
+   }
+   return(complement)
+ }
> # Main program
> binary <- readline("Input an 8-bit binary value: ")
Input an 8-bit binary value: 10100101
> cat("Sample Output:\n")
Sample Output:
> cat("The original binary =", binary, "\n")
The original binary = 10100101
> cat("After ones complement the number =", ones_complement(binary), "\n")
After ones complement the number = 01011010
> 
```

24] Write a program in R to find two's complement of a binary number.
Sample Output:
Input a 8 bit binary value: 01101110
The original binary = 01101110
After one's complement the value = 10010001
After two's complement the value = 10010010

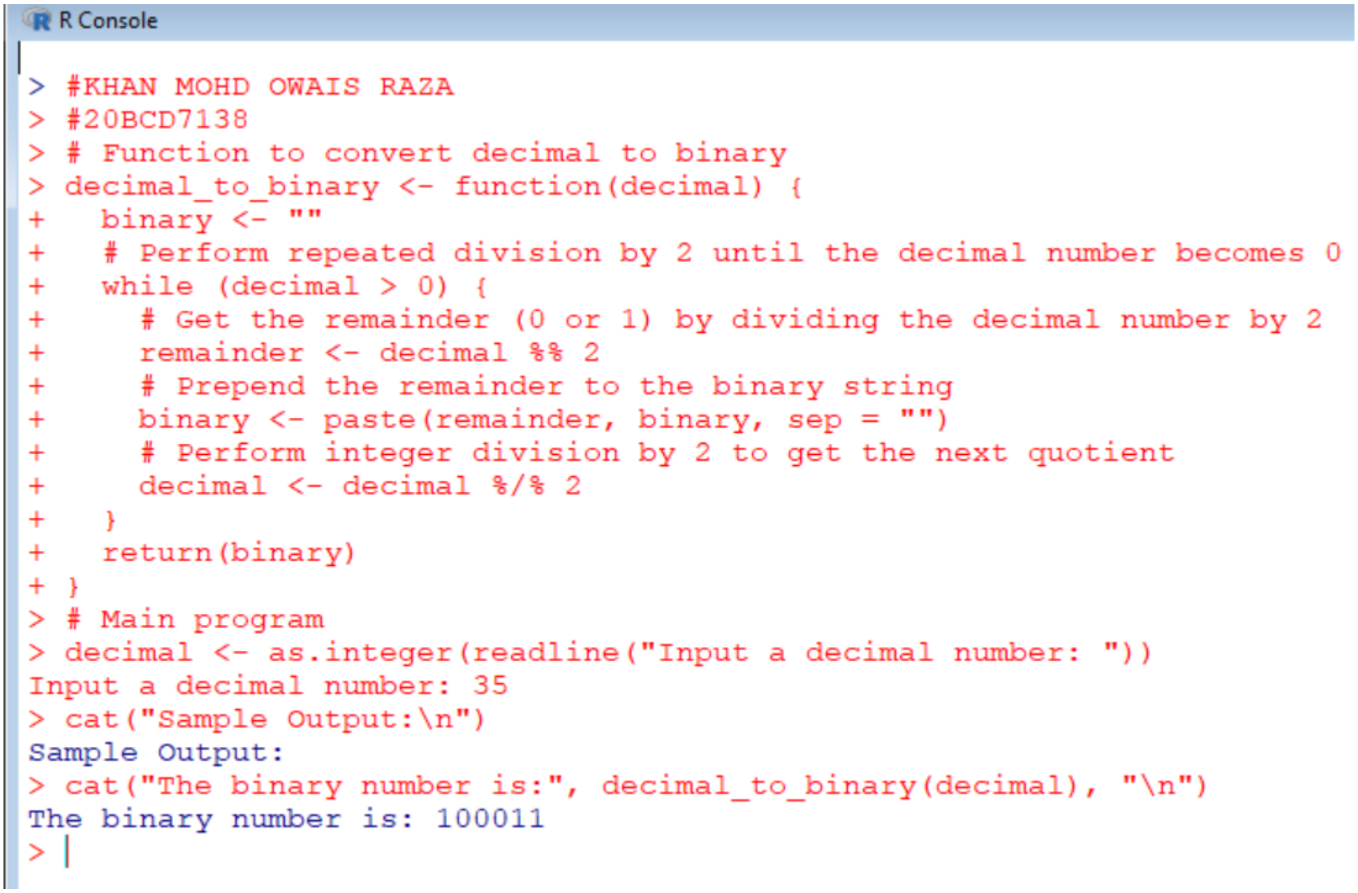
```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to find one's complement of a binary number
ones_complement <- function(binary) {
  complement <- ""
  # Iterate through each bit of the binary number
  for (i in 1:nchar(binary)) {
    bit <- substr(binary, i, i)
    # Invert the bit (0 becomes 1 and 1 becomes 0)
    inverted_bit <- ifelse(bit == "0", "1", "0")
    complement <- paste(complement, inverted_bit, sep = "")
  }
  return(complement)
}
# Function to find two's complement of a binary number
twos_complement <- function(binary) {
  ones_comp <- ones_complement(binary)
  # Add 1 to the least significant bit of the one's complement
  twos_comp <- as.character(as.binary(as.integer(ones_comp, base
= 2) + 1))
  # Pad with leading zeros to maintain the same number of bits
  twos_comp <- sprintf("%08s", twos_comp)
  return(twos_comp)
}
# Main program
binary <- readline("Input an 8-bit binary value: ")
cat("Sample Output:\n")
cat("The original binary =", binary, "\n")
cat("After one's complement the value =",
ones_complement(binary), "\n")
cat("After two's complement the value =",
twos_complement(binary), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to find one's complement of a binary number
> ones_complement <- function(binary) {
+   complement <- ""
+   # Iterate through each bit of the binary number
+   for (i in 1:nchar(binary)) {
+     bit <- substr(binary, i, i)
+     # Invert the bit (0 becomes 1 and 1 becomes 0)
+     inverted_bit <- ifelse(bit == "0", "1", "0")
+     complement <- paste(complement, inverted_bit, sep = "")
+   }
+   return(complement)
+ }
> # Function to find two's complement of a binary number
> twos_complement <- function(binary) {
+   ones_comp <- ones_complement(binary)
+   # Add 1 to the least significant bit of the one's complement
+   twos_comp <- as.character(as.binary(as.integer(ones_comp, base = 2) + 1))
+   # Pad with leading zeros to maintain the same number of bits
+   twos_comp <- sprintf("%08s", twos_comp)
+   return(twos_comp)
+ }
> # Main program
> binary <- readline("Input an 8-bit binary value: ")
Input an 8-bit binary value: 01101110
> cat("Sample Output:\n")
Sample Output:
> cat("The original binary =", binary, "\n")
The original binary = 01101110
```

25] Write a program in R to convert a decimal number to a binary number.
Sample Output:
Input a decimal number: 35
The binary number is: 100011

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Function to convert decimal to binary
decimal_to_binary <- function(decimal) {
  binary <- ""
  # Perform repeated division by 2 until the decimal number
becomes 0
  while (decimal > 0) {
    # Get the remainder (0 or 1) by dividing the decimal number
by 2
    remainder <- decimal %% 2
    # Prepend the remainder to the binary string
    binary <- paste(remainder, binary, sep = "")
    # Perform integer division by 2 to get the next quotient
    decimal <- decimal %/% 2
  }
  return(binary)
}
# Main program
decimal <- as.integer(readline("Input a decimal number: "))
cat("Sample Output:\n")
cat("The binary number is:", decimal_to_binary(decimal), "\n")
```



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Function to convert decimal to binary
> decimal_to_binary <- function(decimal) {
+   binary <- ""
+   # Perform repeated division by 2 until the decimal number becomes 0
+   while (decimal > 0) {
+     # Get the remainder (0 or 1) by dividing the decimal number by 2
+     remainder <- decimal %% 2
+     # Prepend the remainder to the binary string
+     binary <- paste(remainder, binary, sep = "")
+     # Perform integer division by 2 to get the next quotient
+     decimal <- decimal %/% 2
+   }
+   return(binary)
+ }
> # Main program
> decimal <- as.integer(readline("Input a decimal number: "))
Input a decimal number: 35
> cat("Sample Output:\n")
Sample Output:
> cat("The binary number is:", decimal_to_binary(decimal), "\n")
The binary number is: 100011
> 
```