1. **Write a R program to take input from theuser (name and age) and display the values.**

Program:

```
name <- readline(prompt = "Enter your name: ")
age <- as.integer(readline(prompt = "Enter your age: "))

cat("Name:", name, "\n")
cat("Age:", age, "\n")

cat("R Version:", R.version.string, "\n")
```

output:

Enter your name: John
Enter your age: 25
Name: John
Age: 25
R Version: R version 4.1.2 (2021-11-01) -- "Bird Hippie"

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

Program:

```
x <- 5
y <- c(1, 2, 3, 4, 5)
z <- matrix(1:9, nrow = 3)
cat("Objects in memory:\n")
print(ls())
cat("\nStructure of each object:\n")
str(x)
str(y)
str(z)
cat("\nMemory size of each object:\n")
cat("Size of x:", object.size(x), "bytes\n")
cat("Size of y:", object.size(y), "bytes\n")
```

```
cat("Size of z:", object.size(z), "bytes\n")
cat("\nTotal memory used by R (Windows only):\n")
cat("Memory used:", memory.size(), "bytes\n")
```

output:

```
Objects in memory:
[1] "x" "y" "z"

Structure of each object:
num 5
 num [1:5] 1 2 3 4 5
 int [1:3, 1:3] 1 2 3 4 5 6 7 8 9

Memory size of each object:
Size of x: 40 bytes
Size of y: 112 bytes
Size of z: 120 bytes

Total memory used by R (Windows only):
Memory used: 3456 bytes
```

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 ofnumbers from 51 to 91.

Program:

```
sequence_20_50 <- 20:50
cat("Sequence from 20 to 50:\n")
print(sequence_20_50)
numbers_20_60 <- 20:60
mean_20_60 <- mean(numbers_20_60)
cat("\nMean of numbers from 20 to 60:\n")
print(mean_20_60)
numbers_51_91 <- 51:91
sum_51_91 <- sum(numbers_51_91)
cat("\nSum of numbers from 51 to 91:\n")
print(sum_51_91)
```

output:

Sequence 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

Mean of numbers from 20 to 60:

40

Sum of numbers from 51 to 91:

1716

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

Program:

```
random_integers <- sample(-50:50, 10, replace =TRUE)
cat("Random integers between -50 and +50:\n")
print(random_integers)
```

output:

Random integers between -50 and +50:
[1] 7 -24 15 4 30 -42 47 21 -6 0

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

Program:

```
fibonacci <- function(n) {
```

```
 fib <- numeric(n)  # Create an empty numeric vector to store Fibonacci numbers
 fib[1] <- 0  # First Fibonacci number
 fib[2] <- 1  # Second Fibonacci number

 for (i in 3:n) {
   fib[i] <- fib[i - 1] + fib[i - 2]  # Fibonacci relation
 }

 return(fib)  # Return the Fibonacci sequence
}
first_10_fibonacci <- fibonacci(10)
cat("First 10 Fibonacci numbers:\n")
print(first_10_fibonacci)
```

output:

First 10 Fibonacci numbers: [1] 0 1 1 2 3 5 8 13 21 34

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

Program:

```
sieve_of_eratosthenes <- function(n) {
 primes <- rep(TRUE, n + 1)  # True means prime, false means not prime
 primes[1] <- FALSE  # 1 is not a prime
 for (i in 2:sqrt(n)) {
  if (primes[i]) {
   primes[seq(i * i, n, by = i)] <- FALSE
  }
 }
```

```
}
n <- as.integer(readline(prompt = "Enter the number up to which you want to find
primes: "))
prime_numbers <- sieve_of_eratosthenes(n)
cat("Prime numbers up to", n, "are:", prime_numbers, "\n")
```

output:

Enter the number up to which you want to find primes: 30
Prime numbers up to 30 are: 2 3 5 7 11 13 17 19 23 29

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

Program:

```
fizz_buzz <- function() {
 # Loop through numbers from 1 to 100
 for (i in 1:100) {
  if (i %% 3 == 0 & i %% 5 == 0) {
   cat("FizzBuzz\n")
  } else if (i %% 3 == 0) {
   cat("Fizz\n")
  } else if (i %% 5 == 0) {
   cat("Buzz\n")
  } else {
   cat(i, "\n")
  }
 }
}
```

fizz_buzz()

output:

1

2

Fizz

4

Buzz

Fizz

7

8

Fizz

Buzz

11

Fizz

13

14

FizzBuzz

16

17

Fizz

19

Buzz

Fizz

22

23

Fizz

Buzz

26

Fizz

28

29

FizzBuzz

31

32

Fizz

34

Buzz

Fizz

37

38

Fizz

Buzz

41

Fizz

43

44

FizzBuzz

46

47

Fizz

49

Buzz

Fizz

52

53

Fizz

Buzz

56

Fizz

58

59

FizzBuzz

61

62

Fizz

64

Buzz

Fizz

67

68

Fizz

Buzz

71

Fizz

73

74

FizzBuzz

76

77

Fizz

79

Buzz

Fizz

82

83

Fizz

Buzz

86

Fizz

88

89

FizzBuzz

91

92

Fizz

94

Buzz

Fizz

97

98

Fizz

Buzz

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

Program:

<- function() {

```
 letters_lower <- letters

 first_10_lower <- letters_lower[1:10]

 last_10_upper <- toupper(letters_lower[17:26])

   cat("First 10 English letters in lowercase: ", paste(first_10_lower, collapse = ""), "\n")
 cat("Last 10 English letters in uppercase: ", paste(last_10_upper, collapse = ""), "\n")
 cat("Letters between 22nd and 24th in uppercase: ", paste(between_22_24_upper, collapse
= ""), "\n")
}

extract_letters()
```

output:

First 10 English letters in lowercase: abcdefghij

Last 10 English letters in uppercase: QRSTUVWXYZ

Letters between 22nd and 24th in uppercase: VWX

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

Program:

```
find_factors <- function(n) {
  factors <- c()  # Create an empty vector to store factors
```

```
 for (i in 1:n) {
  if (n %% i == 0) {  # If n is divisible by i
   factors <- c(factors, i)  # Add i to the factors vector
  }
 }
 return(factors)
}

number <- as.integer(readline(prompt = "Enter a number to find its factors: "))

factors <- find_factors(number)


cat("The factors of", number, "are:", factors, "\n")
```

output:

Enter a number to find its factors: 36
The factors of 36 are: 1 2 3 4 6 9 12 18 36

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

Program:

```
find_max_min <- function(vec) {
 max_value <- max(vec)  # Find the maximum value in the vector
 min_value <- min(vec)  # Find the minimum value in the vector
 return(list(max_value = max_value, min_value = min_value))
}vec_input <- as.numeric(unlist(strsplit(readline(prompt = "Enter a vector of numbers
(separated by spaces): "), " ")))

# Get the maximum and minimum values using the function
result <- find_max_min(vec_input)
cat("The maximum value of the vector is:", result$max_value, "\n")
cat("The minimum value of the vector is:", result$min_value, "\n")
```

output:

Enter a vector of numbers (separated by spaces): 4 7 2 9 5 1
The maximum value of the vector is: 9
The minimum value of the vector is: 1

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

Program:

```
unique_characters <- function(input_string) {
 unique_chars <- unique(strsplit(input_string, NULL)[[1]])
 return(unique_chars)
}

unique_numbers <- function(input_vector) {
 unique_vals <- unique(input_vector)
 return(unique_vals)
}

input_string <- "programming"
input_vector <- c(1, 2, 3, 2, 4, 5, 1, 6, 4)
unique_chars <- unique_characters(input_string)
unique_vals <- unique_numbers(input_vector)
cat("Unique characters in the string: ", unique_chars, "\n")
cat("Unique numbers in the vector: ", unique_vals, "\n")
```

output:

Unique characters in the string: p r o g a m i ng

Unique numbers in the vector: 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.

Program:

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

b <- c(4, 5, 6)

c <- c(7, 8, 9)

matrix_combined <- cbind(a, b, c)


cat("The 3x3 matrix is:\n")

print(matrix_combined)
```

output:

The 3x3 matrix is:

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.

Program:

```
set.seed(123)


random_numbers <- rnorm(100, mean = 0, sd = 1)
```

```
rounded_numbers <- round(random_numbers, 2)

occurrences <- table(rounded_numbers)

cat("Occurrences of each value in the generated list:\n")
print(occurrences)
```

output:

Occurrences of each value in the generated list:

rounded_numbers

-2.69 -2.49 -2.42 -2.33 -2.19 -2.14 -2.06 -1.92 -1.86 -1.74 -1.70 -1.66 -1.61 -1.55 -1.44 -1.39 -1.34 -1.32 -1.21 -1.18 -1.11 -1.04 -1.02 -0.99 -0.97 -0.90 -0.83 -0.81 -0.76 -0.69 -0.63 -0.60 -0.53 -0.48 -0.47 -0.43 -0.42 -0.38 -0.36 -0.33 -0.30 -0.27 -0.25 -0.23 -0.20 -0.18 -0.17 -0.14 -0.10 -0.09 -0.08 -0.06 -0.05 0.03 0.04 0.06 0.08 0.09 0.11 0.13 0.14 0.17 0.20 0.21 0.23 0.24 0.27 0.29 0.30 0.33 0.36 0.39 0.41 0.43 0.47 0.50 0.52 0.55 0.57 0.60 0.63 0.64 0.66 0.68 0.72 0.74 0.77 0.79 0.80 0.83 0.85 0.87 0.89 0.90 0.92 0.93 0.95 0.97 0.99 1.02 1.04 1.06 1.08 1.10 1.11 1.13 1.14 1.15 1.17 1.18 1.20 1.21 1.24 1.26 1.28 1.30 1.32 1.34 1.37 1.39 1.42 1.43 1.45 1.47 1.49 1.51 1.53 1.55 1.56 1.58 1.61 1.63 1.65 1.67 1.70 1.73 1.74 1.76 1.79 1.80 1.81 1.82 1.85 1.87 1.88 1.89 1.91 1.93 1.95 1.96 1.97 1.99 2.00 2.02 2.03 2.05 2.07

14.Write R program to read the .csv file and display the content.

Program:

```
install.packages("readr")

library(readr)
data <- read_csv("yourfile.csv")
```

```
print(data)
```

Replace "yourfile.csv" with the path to your actual .csv file.

output:

Name Age Major

<chr> <dbl> <chr>

1 John 20 Computer Science

2 Alice 22 Mathematics

3 Bob 21 Engineering

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.

Program:

```
numeric_vector <- c(1, 2, 3, 4, 5)

character_vector <- c("apple", "banana", "cherry", "date")

logical_vector <- c(TRUE, FALSE, TRUE, FALSE)

cat("Numeric Vector: \n")

print(numeric_vector)

cat("Type of Numeric Vector: ", typeof(numeric_vector), "\n\n")


cat("Character Vector: \n")

print(character_vector)

cat("Type of Character Vector: ", typeof(character_vector), "\n\n")
```

```
cat("Logical Vector: \n")
print(logical_vector)
cat("Type of Logical Vector: ", typeof(logical_vector), "\n")
```

Output:

Numeric Vector:

[1] 1 2 3 4 5

Type of Numeric Vector: double

Character Vector:

[1] "apple" "banana" "cherry" "date"

Type of Character Vector: character

Logical Vector:

[1] TRUE FALSE TRUE FALSE

Type of Logical Vector: 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.

Program:

```
matrix_5x4 <- matrix(1:20, nrow = 5, ncol = 4, byrow = TRUE)
cat("5x4 Matrix (Filled by rows):\n")
print(matrix_5x4)
matrix_3x3 <- matrix(1:9, nrow = 3, ncol = 3, byrow = TRUE)
rownames(matrix_3x3) <- c("Row1", "Row2", "Row3")
colnames(matrix_3x3) <- c("Col1", "Col2", "Col3")
```

```
cat("\n3x3 Matrix with Labels (Filled by rows):\n")
print(matrix_3x3)
matrix_2x2 <- matrix(1:4, nrow = 2, ncol = 2, byrow = FALSE)
rownames(matrix_2x2) <- c("R1", "R2")
colnames(matrix_2x2) <- c("C1", "C2")
cat("\n2x2 Matrix with Labels (Filled by columns):\n")
print(matrix_2x2)
```

output :

5x4 Matrix (Filled by rows):

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
[5,]   17   18   19   20
```

3x3 Matrix with Labels (Filled by rows):

```
     Col1 Col2 Col3
Row1    1    2    3
Row2    4    5    6
Row3    7    8    9
```

2x2 Matrix with Labels (Filled by columns):

```
   C1 C2
R1  1  3
R2  2  4
```

17.Write a R program to create an array, passing in a vector of values and a vector of dimensions. Also provide names for each dimension.

Program:

```
values <- 1:12

dim_values <- c(3, 2, 2)

my_array <- array(values, dim = dim_values)

dimnames(my_array) <- list(Row = c("R1", "R2", "R3"),
                           Column = c("C1", "C2"),
                           Depth = c("D1", "D2"))


print(my_array)
```

output :

```
: , , D1

   Column
Row  C1 C2
 R1   1  4
 R2   2  5
 R3   3  6

, , D2

   Column
Row  C1 C2
 R1   7 10
 R2   8 11
 R3   9 12
```

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

program:

```
vector1 <- c(1, 2, 3, 4, 5, 6, 7, 8, 9)

vector2 <- c(10, 11, 12, 13, 14, 15, 16, 17, 18)

combined_vector <- c(vector1, vector2)

dim_values <- c(3, 3, 2)

my_array <- array(combined_vector, dim = dim_values)

print(my_array)
```

output :

```
, , 1

     [,1] [,2] [,3]

[1,]    1    4    7

[2,]    2    5    8

[3,]    3    6    9

, , 2

     [,1] [,2] [,3]

[1,]   10   13   16

[2,]   11   14   17

[3,]   12   15   18
```

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

Program:

```
my_vector <- c(1, 2, 3, 4, 5)
```

```
my_matrix <- matrix(1:9, nrow = 3, ncol = 3)
my_function <- function(x) {
 return(x^2)
}
my_list <- list(Vector = my_vector,
          Matrix = my_matrix,
          Function = my_function)
print(my_list)
result <- my_list$Function(3)
cat("Function result (3^2):", result, "\n")
```

output :

```
$Vector
[1] 1 2 3 4 5
$Matrix
   [,1] [,2] [,3]
[1,]   1   4   7
[2,]   2   5   8
[3,]   3   6   9
$Function
function(x) {
 return(x^2)
}
Function result (3^2): 9
```

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

Program:

20.

```
plot(NA, xlim = c(0, 10), ylim = c(0, 20),
    xlab = "X Axis", ylab = "Y Axis", main = "Empty Plot with Axis Limits")
```

output :

When you run this code in an R environment, it will display a plot with:

The x-axis ranging from 0 to 10.

The y-axis ranging from 0 to 20.

No points plotted (since the plot is empty, with NA as input).

The labels "X Axis" and "Y Axis" for the respective axes.

The title "Empty Plot with Axis Limits".