

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.

The screenshot shows the RStudio interface. The script editor contains the following code:

```
1 name<-as.character(readline("Enter the name:"))
2 age<-as.integer(readline("Enter the age:"))
3 cat(name,age)
4 cat("\n R Version: ",R.version.string)
5
```

The console shows the output of the script:

```
R 4.3.2 ~-~
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> name<-as.character(readline("Enter the name:"))
Enter the name:Sai
> age<-as.integer(readline("Enter the age:"))
Enter the age:19
> cat(name,age)
Sai 19
> cat("\n R Version: ",R.version.string)
R Version: R version 4.3.2 (2023-10-31 ucrt)
>
```

The Environment pane on the right shows the values of the variables:

Values	
age	19L
name	"Sai"

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

The screenshot shows the RStudio interface. The script editor contains the following code:

```
1 name = "python";
2 n1 = 10;
3 n2 = 0.5
4 nums = c(10, 20, 30, 40, 50, 60)
5 print(ls())
6 print("Details of the objects in memory:")
7 print(ls.str())
```

The console shows the output of the script:

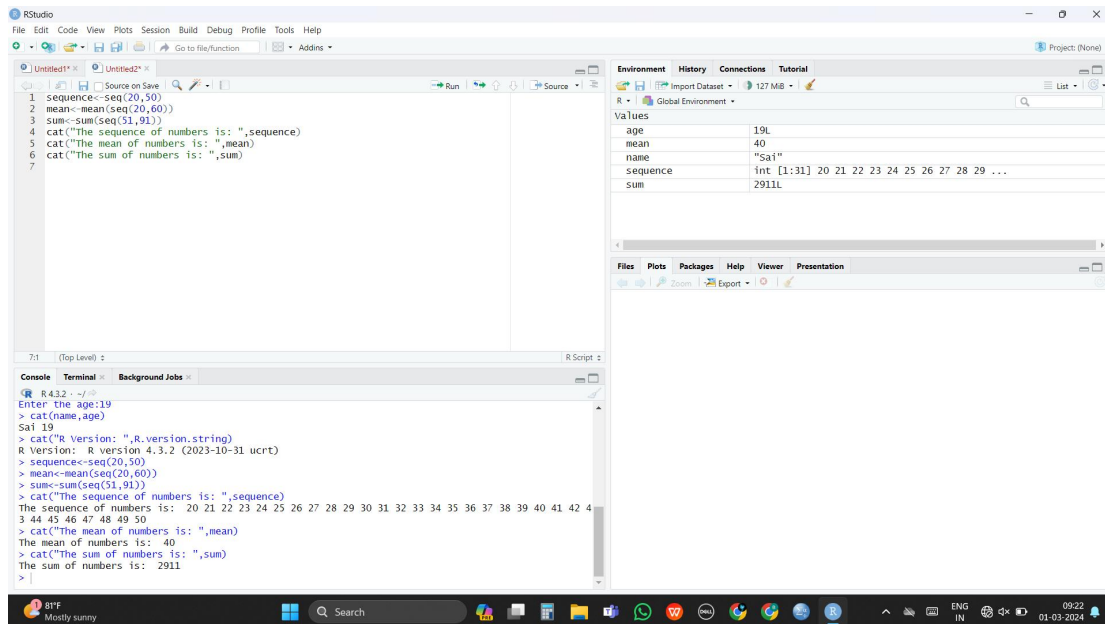
```
R 4.3.2 ~-~
Type 'q()' to quit R.

> name = "python";
> n1 = 10;
> n2 = 0.5
> nums = c(10, 20, 30, 40, 50, 60)
> print(ls())
[1] "n1" "n2" "name" "nums"
> print("Details of the objects in memory:")
[1] "Details of the objects in memory:"
> print(ls.str())
n1 : num 10
n2 : num 0.5
name : chr "python"
nums : num [1:6] 10 20 30 40 50 60
>
```

The Environment pane on the right shows the values of the objects:

Values	
n1	10
n2	0.5
name	"python"
nums	num [1:6] 10 20 30 40 50 60

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



The screenshot shows the RStudio interface. The script editor contains the following code:

```
1 sequence<-seq(20,50)
2 mean<-mean(seq(20,60))
3 sum<-sum(seq(51,91))
4 cat("The sequence of numbers is: ",sequence)
5 cat("The mean of numbers is: ",mean)
6 cat("The sum of numbers is: ",sum)
7
```

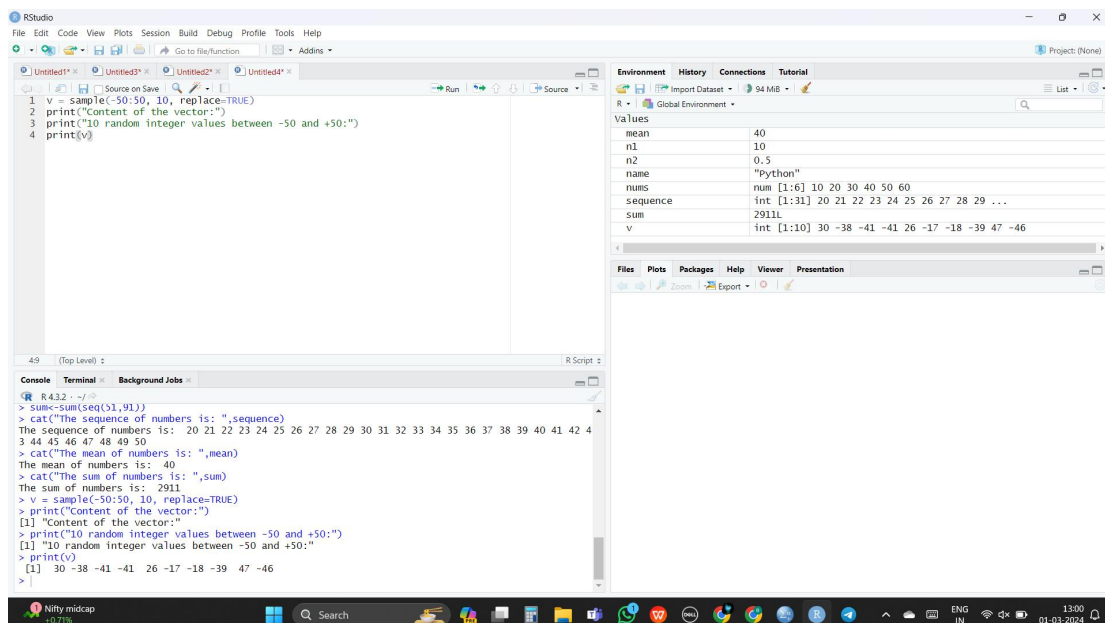
The console shows the output of the script:

```
R 4.3.2 ~ /
Enter the age:19
> cat(name,age)
Sal 19
> cat("R Version: ", R.version.string)
R Version: R version 4.3.2 (2023-10-31 ucrt)
> sequence<-seq(20,50)
> mean<-mean(seq(20,60))
> sum<-sum(seq(51,91))
> cat("The sequence of numbers is: ",sequence)
The sequence of numbers is: 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
> cat("The mean of numbers is: ",mean)
The mean of numbers is: 40
> cat("The sum of numbers is: ",sum)
The sum of numbers is: 2911
>
```

The Environment pane on the right shows the following variables:

Variable	Value
age	19L
mean	40
name	"Sal"
sequence	int [1:31] 20 21 22 23 24 25 26 27 28 29 ...
sum	2911L

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



The screenshot shows the RStudio interface. The script editor contains the following code:

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

The console shows the output of the script:

```
R 4.3.2 ~ /
> sum<-sum(seq(51,91))
> cat("The sequence of numbers is: ",sequence)
The sequence of numbers is: 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
> cat("The mean of numbers is: ",mean)
The mean of numbers is: 40
> cat("The sum of numbers is: ",sum)
The sum of numbers is: 2911
> v = sample(-50:50, 10, replace=TRUE)
> print("Content of the vector:")
[1] "Content of the vector:"
> print("10 random integer values between -50 and +50:")
[1] "10 random integer values between -50 and +50:"
> print(v)
[1] 30 -38 -41 -41 26 -17 -18 -39 47 -46
>
```

The Environment pane on the right shows the following variables:

Variable	Value
mean	40
n1	10
n2	0.5
name	"python"
num	int [1:6] 10 20 30 40 50 60
sequence	int [1:31] 20 21 22 23 24 25 26 27 28 29 ...
sum	2911L
v	int [1:10] 30 -38 -41 -41 26 -17 -18 -39 47 -46

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

The screenshot shows the RStudio interface. The script editor contains the following code:

```

1 n<-as.integer(readline(prompt="Enter the number of terms:"))
2
3 fibonacci <- function(n) {
4   if (n == 1) {
5     return(0)
6   } else if (n == 2) {
7     return(1)
8   } else {
9     return(fibonacci(n - 1) + fibonacci(n - 2))
10  }
11 }
12
13 fibonacci_numbers <- sapply(1:n, fibonacci)
14 cat("The first 10 Fibonacci numbers are: ", fibonacci_numbers, "\n")
15
16

```

The console shows the execution of the code:

```

> n<-as.integer(readline(prompt="Enter the number of terms:"))
Enter the number of terms:10
> fibonacci <- function(n)
+ if (n == 1) {
+   return(0)
+ } else if (n == 2) {
+   return(1)
+ } else {
+   return(fibonacci(n - 1) + fibonacci(n - 2))
+ }
+ }
> fibonacci_numbers <- sapply(1:n, fibonacci)
> cat("The first 10 Fibonacci numbers are: ", fibonacci_numbers, "\n")
The first 10 Fibonacci numbers are:  0 1 1 2 3 5 8 13 21 34
>

```

The Environment window shows the following variables:

Variable	Value
fibonacci_numbers	num [1:10] 0 1 1 2 3 5 8 13 21 34
mean	40
n	10L
n1	10
n2	0.5
name	"python"
nums	num [1:6] 10 20 30 40 50 60
sequence	int [1:31] 20 21 22 23 24 25 26 27 28 29 ...
cum	20111

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

The screenshot shows the OneCompiler interface. The code editor contains the following code:

```

1 prime_numbers <- function(n) {
2   if (n >= 2) {
3     x = seq(2, n)
4     prime_nums = c()
5     for (i in seq(2, n)) {
6       if (any(x == i)) {
7         prime_nums = c(prime_nums, i)
8         x = c(x[(x % i) != 0], i)
9       }
10    }
11    return(prime_nums)
12  }
13  else {
14    stop("Input number should be at least 2.")
15  }
16 }
17
18 prime_numbers(12)

```

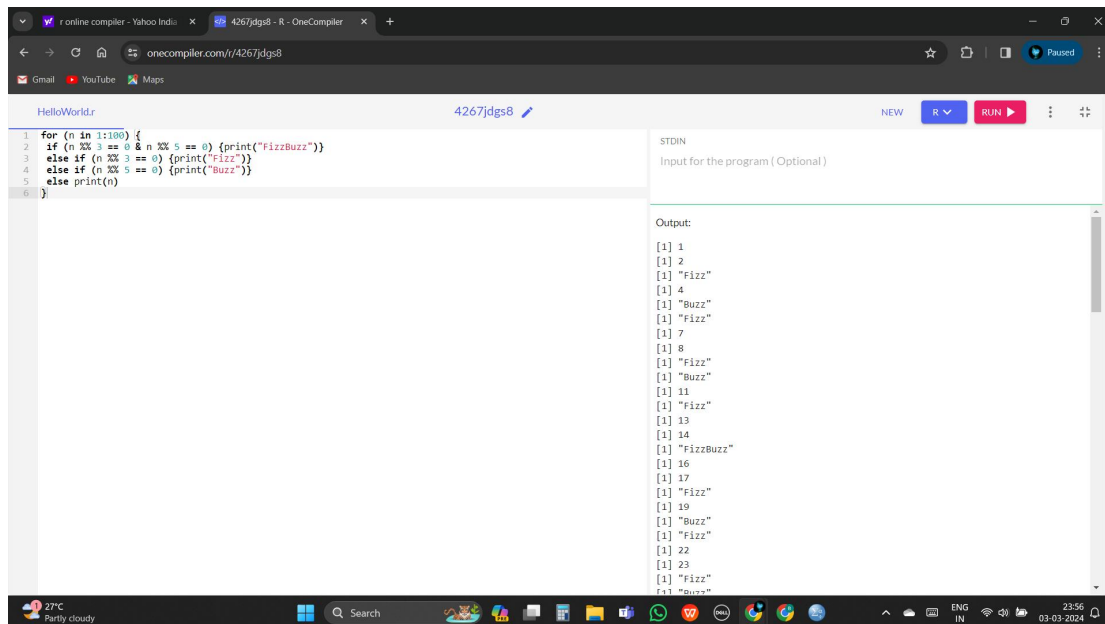
The console shows the execution of the code:

```

[1] 2 3 5 7 11

```

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.



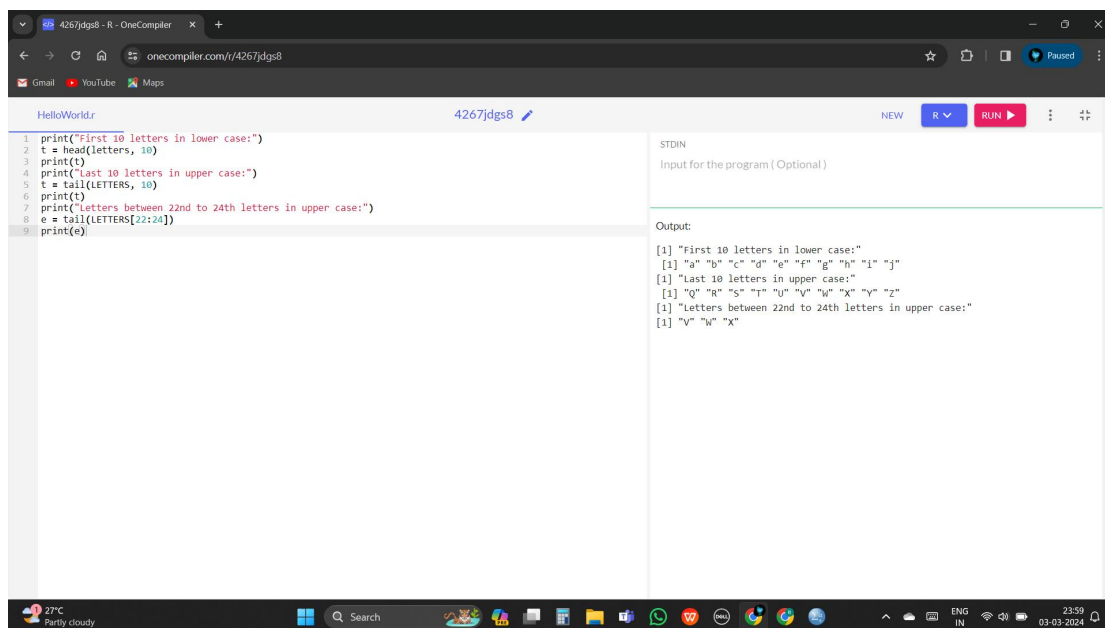
The screenshot shows a web browser with the URL `onecompiler.com/r/4267jds8`. The editor contains the following R code:

```
1 for (n in 1:100) {  
2   if (n %% 3 == 0 & n %% 5 == 0) {print("FizzBuzz")}  
3   else if (n %% 3 == 0) {print("Fizz")}  
4   else if (n %% 5 == 0) {print("Buzz")}  
5   else print(n)  
6 }
```

The output window displays the results of the program:

```
[1] 1  
[1] 2  
[1] "Fizz"  
[1] 4  
[1] "Buzz"  
[1] "Fizz"  
[1] 7  
[1] 8  
[1] "Fizz"  
[1] "Buzz"  
[1] 11  
[1] "Fizz"  
[1] 13  
[1] 14  
[1] "FizzBuzz"  
[1] 16  
[1] 17  
[1] "Fizz"  
[1] 19  
[1] "Buzz"  
[1] "Fizz"  
[1] 22  
[1] 23  
[1] "Fizz"  
[1] "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.



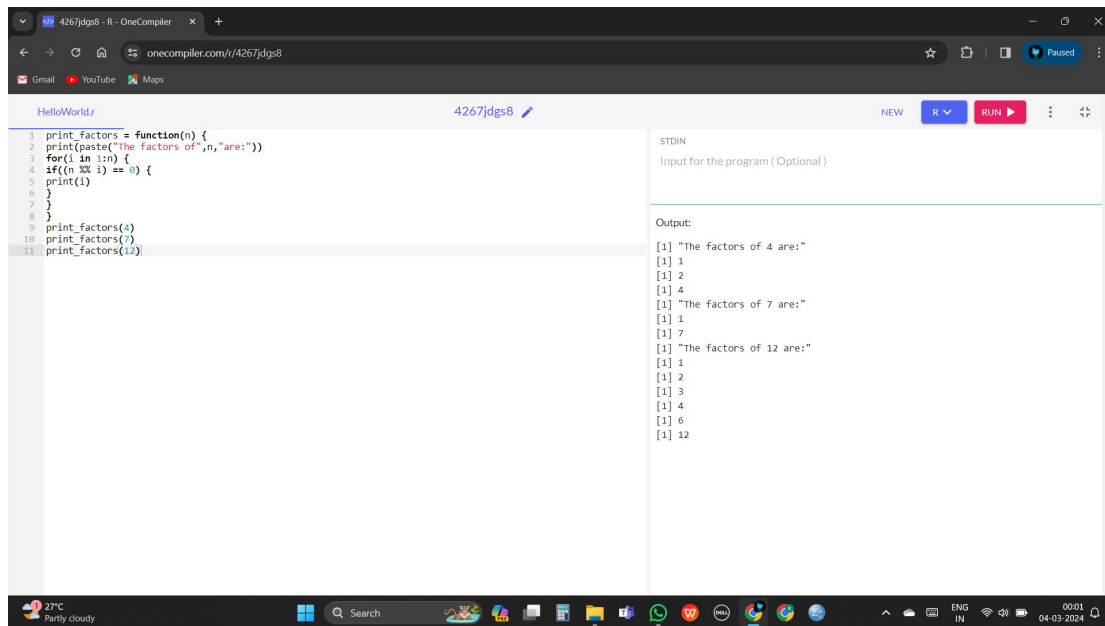
The screenshot shows a web browser with the URL `onecompiler.com/r/4267jds8`. The editor contains the following R code:

```
1 print("First 10 letters in lower case:")  
2 t = head(letters, 10)  
3 print(t)  
4 print("Last 10 letters in upper case:")  
5 t = tail(LETTERS, 10)  
6 print(t)  
7 print("Letters between 22nd to 24th letters in upper case:")  
8 e = tail(LETTERS[22:24])  
9 print(e)
```

The output window displays the results of the program:

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

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



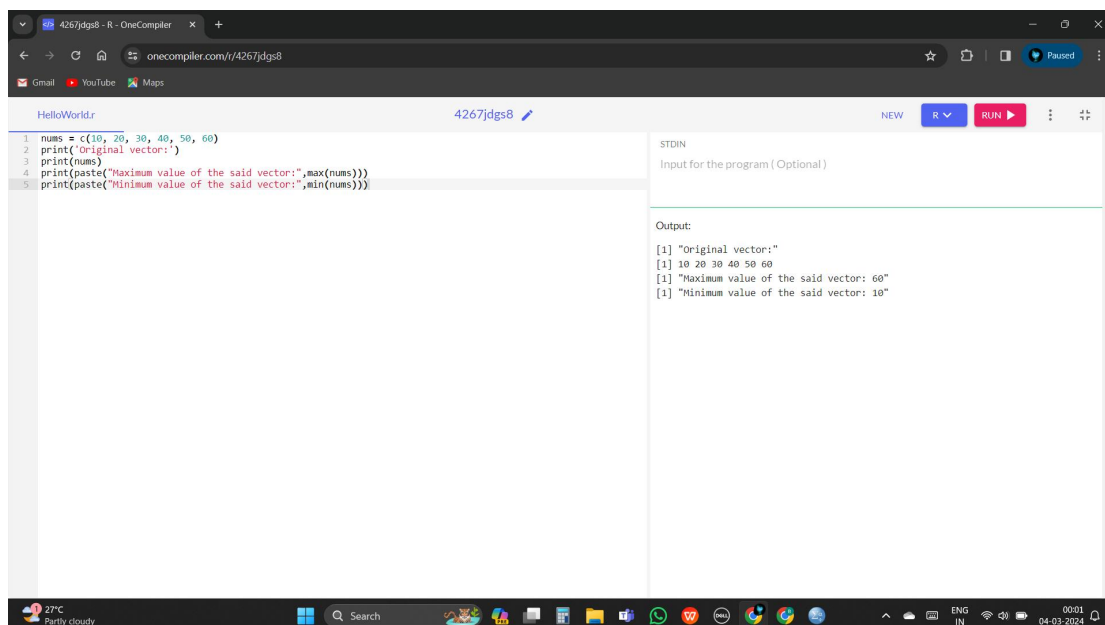
The screenshot shows a web browser window with the OneCompiler website. The editor contains an R script that defines a function `print_factors` to find the factors of a given number `n`. The script then calls this function for the numbers 4, 7, and 12. The output on the right shows the factors for each number: 1 and 2 for 4; 1 and 7 for 7; and 1, 2, 3, 4, 6, and 12 for 12.

```
1 print_factors = function(n) {  
2   print(paste("The factors of",n,"are:"))  
3   for(i in 1:n) {  
4     if((n % i) == 0) {  
5       print(i)  
6     }  
7   }  
8 }  
9 print_factors(4)  
10 print_factors(7)  
11 print_factors(12)
```

Output:

```
[1] "The factors of 4 are:"  
[1] 1  
[1] 2  
[1] 4  
[1] "The factors of 7 are:"  
[1] 1  
[1] 7  
[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



The screenshot shows a web browser window with the OneCompiler website. The editor contains an R script that creates a vector `nums` with the values 10, 20, 30, 40, 50, and 60. It then prints the vector and uses `max` and `min` functions to find the maximum and minimum values. The output on the right shows the original vector and the calculated maximum (60) and minimum (10) values.

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

Output:

```
[1] "Original vector:"  
[1] 10 20 30 40 50 60  
[1] "Maximum value of the said vector: 60"  
[1] "Minimum value of the said vector: 10"
```