**Aim:** Understanding Functions using R

**IDE:** R Studio

**Theory:**

Functions are useful when you want to perform a certain task multiple times. A function accepts input arguments and produces the output by executing valid R commands that are inside the function. In[R Programming Language](https://www.geeksforgeeks.org/r-programming-language-introduction/) when you are creating a function the function name and the file in which you are creating the function need not be the same and you can have one or more functions in R.

### Creating a Function in R

Functions are created in R by using the command function(). The general structure of the function file is as follows:

function\_name <- function(arg\_1, arg\_2, ...) {

Function body

}

## **Function Components**

The different parts of a function are −

* Function Name − This is the actual name of the function. It is stored in R environment as an object with this name.
* Arguments − An argument is a placeholder. When a function is invoked, you pass a value to the argument. Arguments are optional; that is, a function may contain no arguments. Also arguments can have default values.
* Function Body − The function body contains a collection of statements that defines what the function does.
* Return Value − The return value of a function is the last expression in the function body to be evaluated.

## Types of Function in R Language

1. **Built-in Function:**Built-in functions in R are pre-defined functions that are available in [R programming languages](https://www.geeksforgeeks.org/r-programming-language-introduction/) to perform common tasks or operations**.**
2. **User-defined Function:** R language allow us to write our own function.

### Built-in Function in R Programming Language

We will use built-in functions like sum(), max() and min()

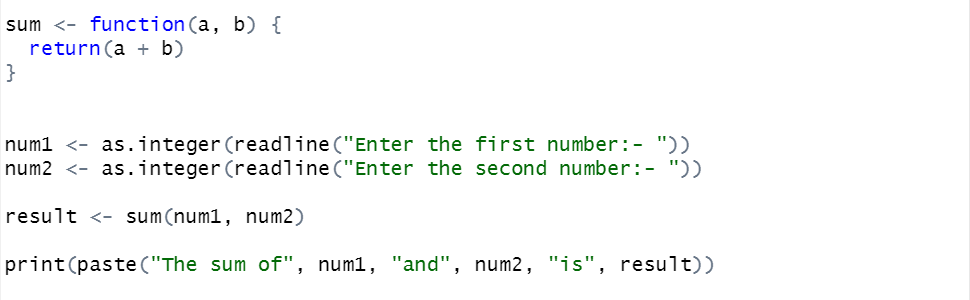
### User-defined Functions in R Programming Language

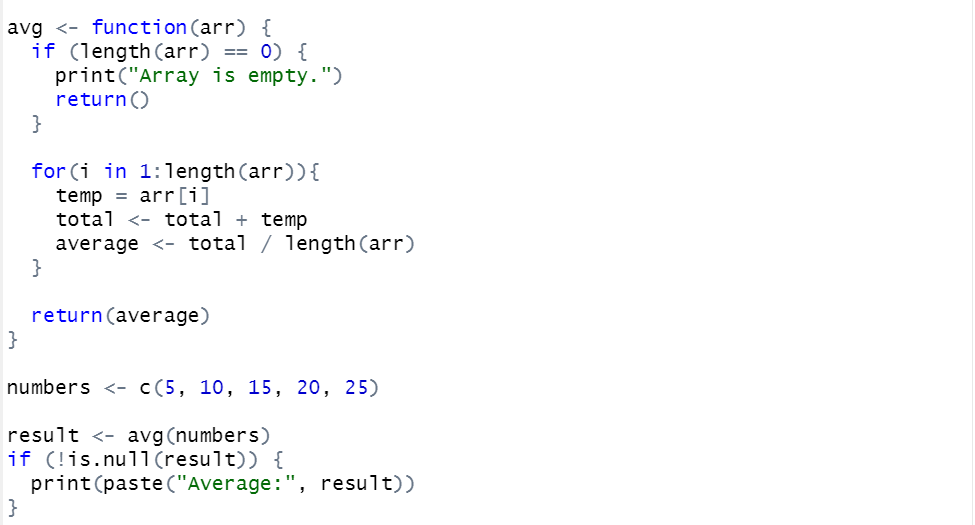
R provides built-in functions like **print()**, **cat()**, etc. but we can also create our own functions. These functions are called user-defined functions.

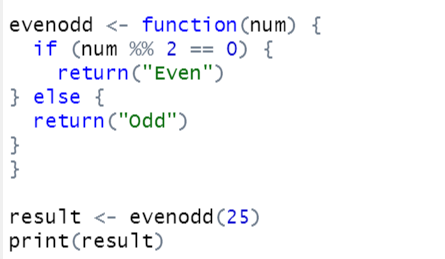
**Programs:**

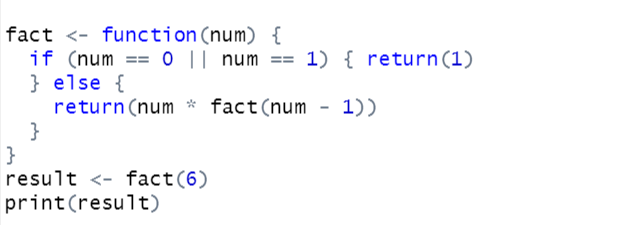
Write R script that demonstrates the working of functions:

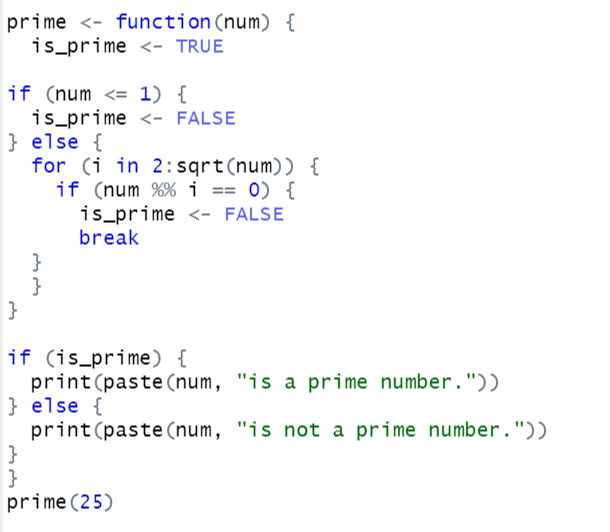
1. Create a user define function named sum which accept 2 arguments (of integer type) and return the sum of them.
2. Create a user define function named avg which pass array as argument (of integer type) and return its average value.
3. Create a user define function named evenodd which accept one argument (of integer type) and return if the number is even or odd.
4. Create a user define function named fact which accept one argument (of integer type) and return the factorial of given number.
5. Create a user define function named prime which accept one argument (of integer type) and display that the no is prime or not. (no need to return value)
6. Create a user define function named power which accept two arguments (of integer type) (i) base (ii) expon and display the base^expon value.
7. Write a function to count the simple interest
8. Write a function to generate the square of any given number
9. Write a function which should work as a power function
10. Write a function which should work as absolute function
11. Write A R Program to Count the number of digit in a user entered number.
12. Write A R Program That prints the Prime Number in Between the Range given by user.
13. Write A R Program to make A Calculator.

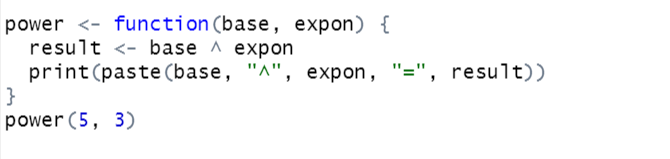


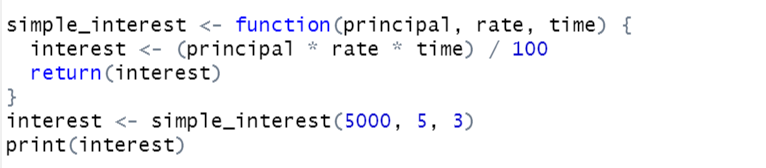


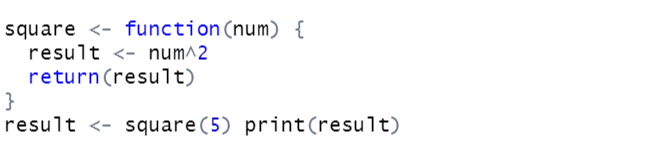
****

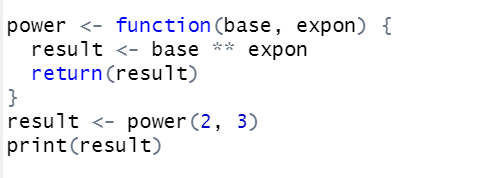
****

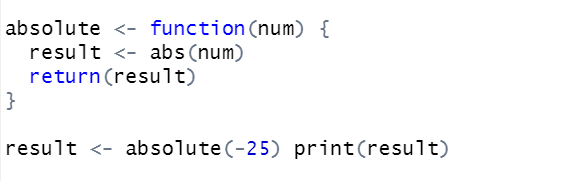
****

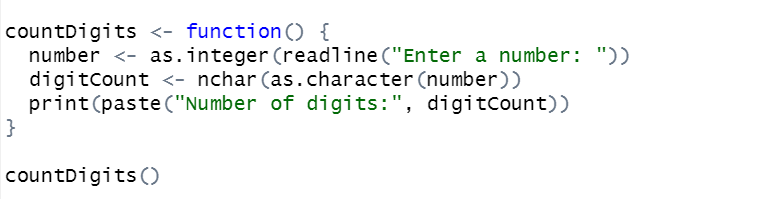
****

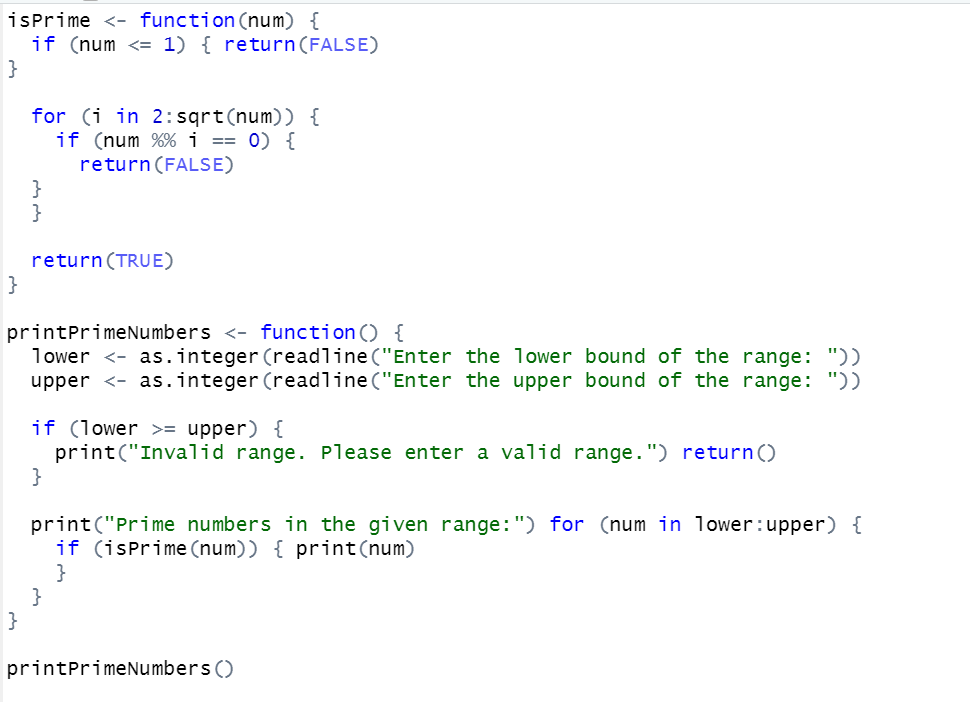
****

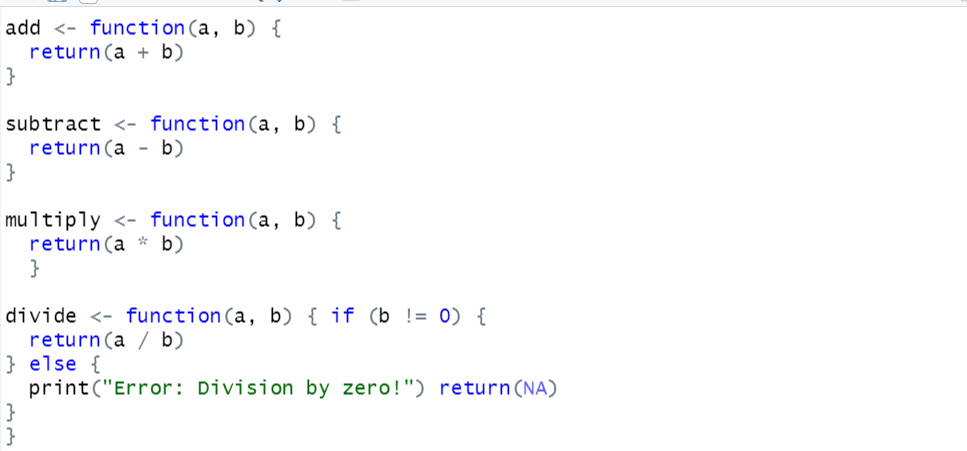
****

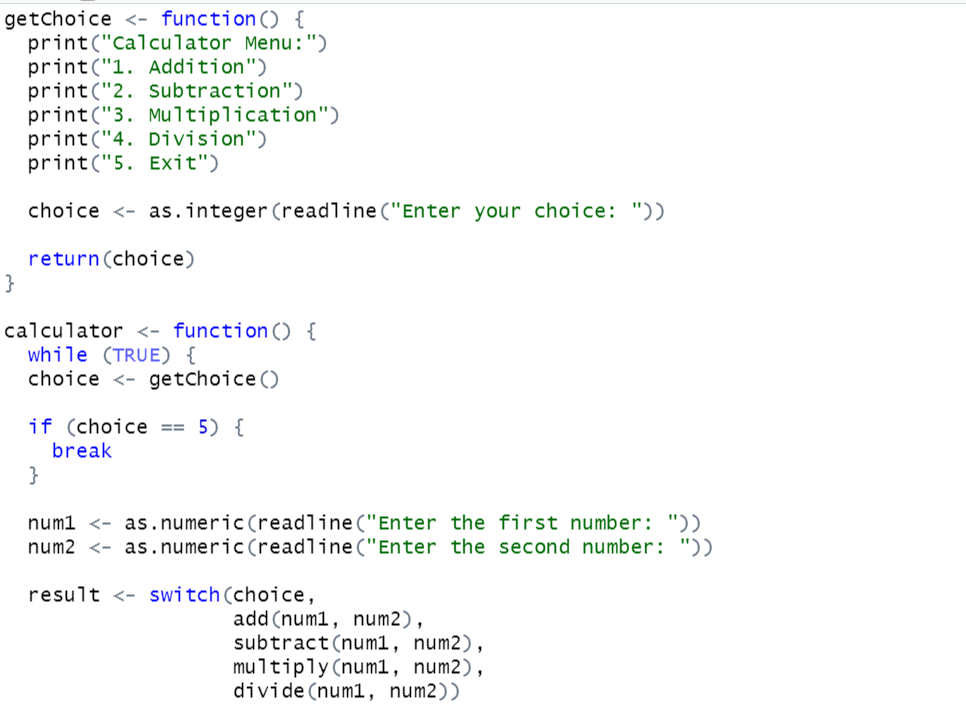
****

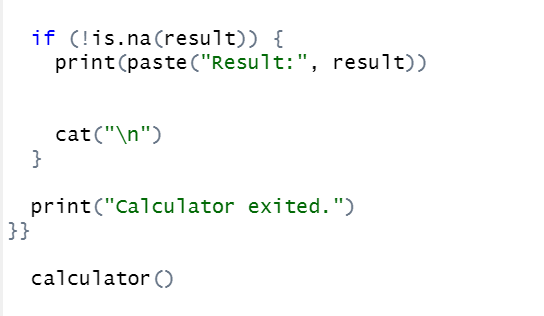
****

****

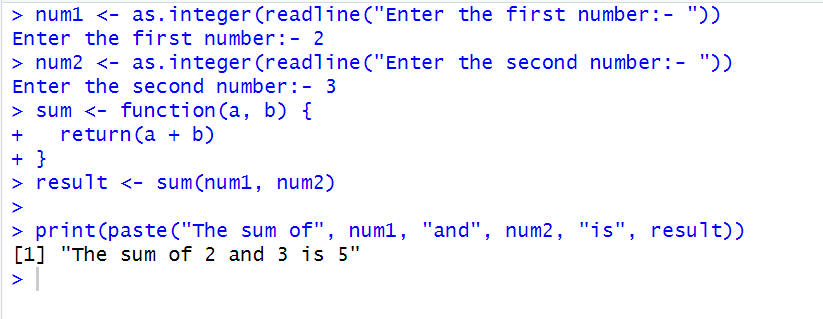
****

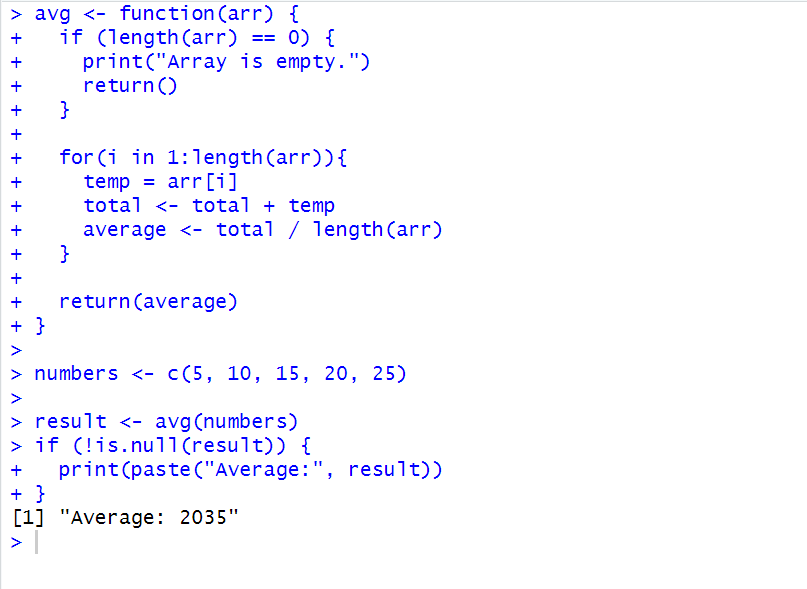
****

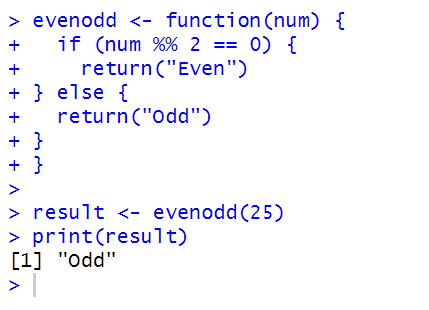
****

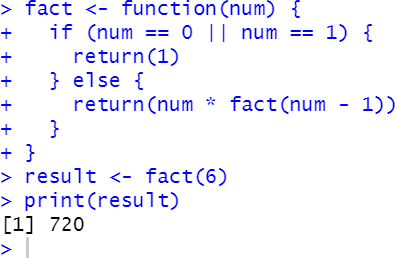
****

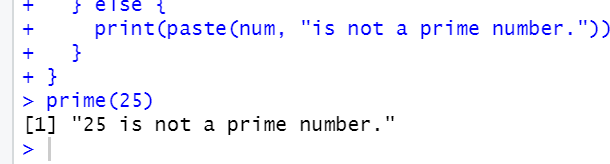
**Output:**

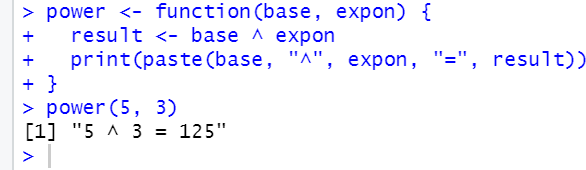
****

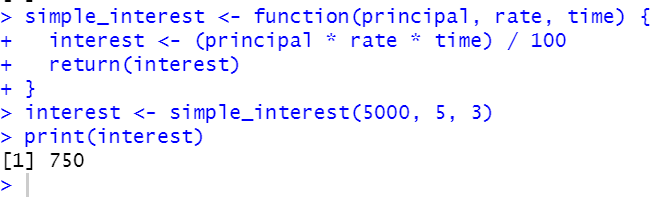
****

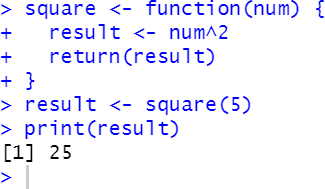
****

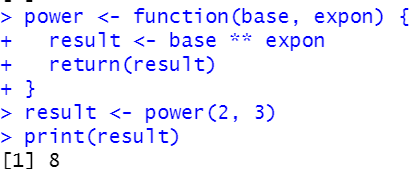


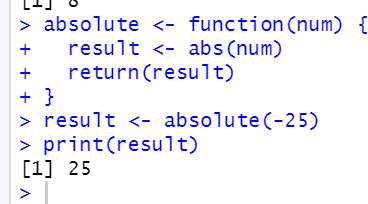


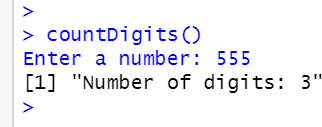


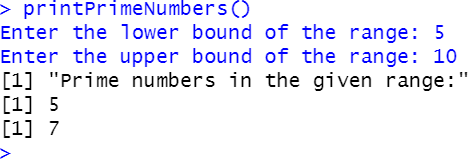


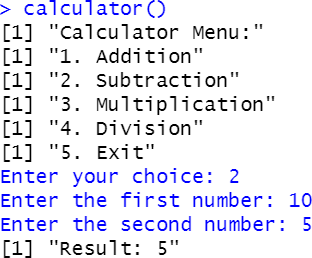












**Observation and Learnings:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.