R

Programming
Part - I

File creation:

```
$ gedit first_program.r
```

Write Code:

```
# My first program in R Programming
Strings <- "Hello, World!"
print ( Strings)
```

Run:

```
$ Rscript first_program.r
```

Output:

```
[1] "Hello, World!"
```

R - Data Types

Vectors Lists Matrices Arrays Factors Data Frames

Vectors		

Logical	TRUE, FALSE	Create a vector:
		fruit <- c('mango','banana',"pineapple")
		print(fruit)

Numeric	3.14, 4, 1000	
Integer	2L, 34L, 0L	
Complex	2+3i	
Character	'A' , "Hello", "TRUE", '13.4'	
Raw	"Hello" is stored as 48 65 6c 6c 6f	

Vectors:

```
> fruit <- c('mango','banana',"pineapple")
> print(fruit)
[1] "mango"    "banana"    "pineapple"
>

> element <- c(2,3,4,1,7,8)
> print(element)
[1] 2 3 4 1 7 8
>
```

Lists:

```
> L <- list(c(10,20,30),3.14,sin)

> print(L)

[[1]]

[1] 10 20 30

[[2]]

[1] 3.14

[[3]]

function (x)  .Primitive("sin")
```

Matrix:

```
> M = matrix( c('x','y','z','y','z','x'), nrow = 2, ncol = 3, byrow = TRUE)
> print(M)
     [,1] [,2] [,3]
[1,] "x"  "y"  "z"
[2,] "y"  "z"  "x"
>
```

Array:

```
> a <- array(c(0,1),dim = c(3,3))
> print(a)
     [,1] [,2] [,3]
[1,]    0    1    0
[2,]    1    0    1
[3,]    0    1    0
>


> a <- array(c(0,1),dim = c(3,3,2))
> print(a)
, , 1

     [,1] [,2] [,3]
[1,]    0    1    0
[2,]    1    0    1
[3,]    0    1    0

, , 2

     [,1] [,2] [,3]
[1,]    1    0    1
[2,]    0    1    0
[3,]    1    0    1
>
```

Factors:

```
> fruit<-c('pineapple','mango','banana','mango','pineapple','mango','banana')
> factor_object_fruit <- factor(fruit)
> print(factor_object_fruit)

[1] pineapple mango     banana    mango     pineapple mango     banana
Levels: banana mango pineapple

> print(nlevels(factor_object_fruit))
[1] 3
>
```

Data Frame:

```
# Create the data frame.

Record <-       data.frame(

 Name = c("Avik", "Ayan","Sukanta"),

   Roll = c(3, 1, 4),

   Stream = c('BA','BSC','BA'),

   Department = c('Bengali','Math','English')

)

print(Record)

     Name Roll Stream Department

1    Avik    3     BA    Bengali

2    Ayan    1    BSC       Math

3 Sukanta    4     BA    English

>
```

If-else statement:

if (test_expression) {
statement
}
	x <- 5
	if(x > 0){
	print("Positive number")
	}
	else {
	print("Negative number")
	}

Loop statement:

Repeat:

repeat {
statement
}
	x <- 1
	repeat {
	print(x)
	x = x+1
	if (x == 6){
	break
	}
	}

While:

while (test_expression)
{
statement
}
	i <- 1
	while (i < 10) {
	print(i)
	i = i+1
	}

For:

for (val in sequence)
{
statement
}
	i=1
	for (i in 1:10) {
	  i=i+1
	  }
	  print(i)

Functions:

Code:

```
sum <- function(a) {
  sum <- 0
  for(i in 1:a) {
    sum=sum+i

  }
  print(paste("sum=",sum))
}
cat("Enter the value of n: ");
a<-readLines("stdin",n=1);
sum(a)
```

Output:

```
Enter the value of n: 5
[1] "sum= 15"
```

1. Write a program to check a number prime of not.

Program Code

```
# Program to check if the input number is prime or not
# take input from the user
cat("Enter a positive number: ");
num <- readLines("stdin",n=1);
num<-as.integer(num)
flag = 0
# prime numbers are greater than 1
if(num > 1) {
# check for factors
flag = 1
for(i in 2:(num-1)) {
if ((num %% i) == 0) {
flag = 0
break
}
}
}
if(num == 2)    flag = 1
if(flag == 1) {
print(paste(num,"is a prime number"))
} else {
print(paste(num,"is not a prime number"))
}
```

```
probir@Incredible:~/Documents/R_program$ Rscript prime_number.R
Enter a positive number: 13
[1] "13 is a prime number"
```

2. Write a program to check the given number arstrong number of not.

Program: Code:

```
# take input from the user
cat("Enter a positive number: ");
num <- readLines("stdin",n=1);
num<-as.integer(num)
count=0
# initialize sum
sum = 0

# find the sum of the cube of each digit
temp = num
while(temp > 0) {
temp = floor(temp / 10)
count=count+1
}
temp=num
```

```
while(temp > 0) {
digit = temp %% 10
sum = sum + (digit ^ count)
temp = floor(temp / 10)
}
# display the result
if(num == sum) {
print(paste(num, "is an Armstrong number"))
} else {
print(paste(num, "is not an Armstrong number"))
}
```

Output:

```
probir@Incredible:~/Documents/R_program$ Rscript amstrong.R
Enter a positive number: 370
[1] "370 is an Armstrong number"
```

3. Write a program to find the factorial of a given number.

Program: Code:

```
facto <- function(){
# accept the input provided by the user and convert to integer
cat("Enter a positive number: ");
num <- readLines("stdin",n=1);
num<-as.integer(num)

fact = 1
# checking whether the number is negative, zero or positive
if(num < 0) {
print(" The number is negative the factorial does not exist. ")
} else if(num == 0) {
print(" The factorial result is 1 ")
} else {
for( i in 1:num) {
fact = fact * i
}
print(paste(" The factorial result is ", num ,"is", fact ))
}
}
facto()
```

Output:

```
Enter a positive number: 5
[1] " The factorial result is  5 is 120"
```

4.Write a program to sort an array in ascending order.

Program Code:

```
#Bubble Sort
sort.b <- function(x)
{
  if(!is.unsorted(x)) {stop("Vector is already sorted")}
  if(length(x)<2){stop("vector is not long enough") }
  if ( !is.vector(x) )  { stop("parameter must be a vector") }
  if ( !is.numeric(x) ) { stop("parameter must be numeric") }

  n = length(x)
  v = x

  for(j in 1:(n-1))
  {
    for(i in 1:(n-j))
    {
      if(v[i+1]<v[i])
      {
        t = v[i+1]
        v[i+1] = v[i]
        v[i] = t
      }
    }
  }
  print(v)
  x = v
}
cat("Enter number of elements: ");
n <- readLines("stdin",n=1);
n<-as.numeric(n)
cat("Enter elements: ");
x<-c()
for(i in 1:n)
{
x[i] <- readLines("stdin",n=1);

i=i+1
}
x<-as.numeric(x)
sort.b(x)
```

Output:

```
Enter number of elements: 5
Enter elements: 3
2
-1
5
1
[1] -1  1  2  3  5
```