Unit 2nd

1. Explain any 5 R data types with example.
2. Explain R looping and looping control stmts with example.
3. Write R program to create pie charts & Bar charts
4. What is Data? Identify and Explain diff categories of Data
5. Write short note on R-arrays & R-factors.
6. Explain the following built-in-functions of R with syntax & example

i) unlist()
ii) rbind()
iii) format()
iv) barplot()

7. Discuss cbind() and rbind() with example
8. Demonstrate any 5 R objects with example

(8) There are many types of R-objects. The frequently used are:

i) Vectors
ii) Lists
iii) Matrices
iv) Arrays
v) Factors
vi) Data frame

i) Vectors:-
When we create vector with more than one element, we use c() function which means to combine elements into a vector.

eg:- apple ← c('red', 'green', 'yellow')
print(apple)

o/p:- "red" "green" "yellow"

ii) Lists:-
A list is an R-object which can contain many diff types of elements.

eg:- list1 ← list(c(2,5,3), 21.3, sin)
print(list1)

o/p:- [[1]]
[1] 2 5 3

[[2]]
[1] 21.3

[[3]]
function (x). Primitive ("sin")

iii) Matrices:-
A matrix is a 2-D rectangular set. it can be created using a vector p/p to the matrix function.

eg:- M = matrix(c('a','a','b','c','b','a'), nrow=2, ncol=3, byrow=TRUE)

o/p:-

	[1]	[2]	[3]
[1]	"a"	"a"	"b"
[2]	"c"	"b"	"a"

iv) Arrays:-
while matrices are confined to 2-D, arrays can be of any Dimensions.

eg:- a ← array(c('green','yellow'), dim=(3,3,2))

o/p:- , , 1

	[1]	[2]	[3]
[1]	"green"	"yellow"	"green"
[2]	"yellow"	"green"	"yellow"
[3]	"green"	"yellow"	"green"

, , 2

	[1]	[2]	[3]
[1]	"yellow"	"green"	"yellow"
[2]	"green"	"yellow"	"green"
[3]	"yellow"	"green"	"yellow"

v) factors:-
factors are the R- objects created using a vector. factor can be created using factor() function.

```
apple_colors ← c('green', 'green', 'yellow', 'red',
                 'red', 'red', 'green')

factor_apple ← factor(apple_colors)
print(factor_apple)
```

o/p:-

[1] green green yellow red red red yellow green

Levels: green red yellow

vi) Data frames:-

Unlike matrices in Data forms each col. can contain dif modes of data.

eg:-

```
BMI ← data.frame(
        gender = c("male", "male", "female"),
        height = c(152, 172, 155),
        weight = c(50, 50, 20),
        Age = c(42, 44, 30).
      )
print(BMI)
```

o/p:-

	gender	height	weight	Age
1	male	152	50	42
2	male	172	50	44
3	female	155	20	30

(2) Looping:-

A loop statement allows us to execute a statement or a group of stmts multiple times.

The given are R-Looping statements with example:-

i) R-Repeat Loop:-

→ The Repeat loop executes the same code again & again until a stop cond^n is met

Syntax:-

```
repeat {
    Commands
    if(cond^n) {
      break
    }
}
```

eg:-

```
v ← c("Hello", "Loop")
cnt ← 2
repeat {
    print(v)
    cnt ← cnt+1
    if(cnt>5){
    break
    }
}
```

o/p:-

"Hello" "Loop"
"Hello" "Loop"
"Hello" "Loop"
"Hello" "Loop"

ii) R- while loop:-
The while loop Repeats stmts or group of stmts while the given Cond" is true it test cond" before executing loop body.

Syntax:-
```
while (test_expression) {
    statements
}
```

eg:- v ← c("Hello" "while loop")
```
cnt ← 0
while (cnt ≤ 4){
    print(v)
    cnt = cnt+1
}
```

o/p - [1] "Hello" "while loop"
"Hello" "while loop"
"Hello" "while loop"
"Hello" "while loop"

iii) R- for loop:-
like a while stmt, it test cond" at the end of the loop body

Syntax:- for (value in vector){
```
    stmts
}
```

eg:- v ← LETTER[1:4]
```
for (i in v){
    print(i)
}
```

o/p:- "A"
"B"
"C"
"D"

Loop Control statements are:-

→ Loop Control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects created in that scope are destroyed.

i) R-Break statements:-
when break stmt is encountered inside a loop, the loop is immediately terminated and Pgm Control goes to next stmt following the loop

```
eg:- v ← c("Hello")
     cnt ← 2
     repeat {
        print(v)
        cnt ← cnt + 1
        if (cnt > 5){
           break
        }
     }
```

o/p:- "Hello"
"Hello"
"Hello"
"Hello"

ii) R-Next statement:-
R-Next stmt is useful when we want to skip the current iteration of a loop

eg:- v ← LETTERS [1:4]
for (i in v) {
if (i == "D") {
next
}
print(i)
}

o/p:- "A"
"B"
"C"
"D"

(5)/(1) Different R-data types are:-

i) Logical:- TRUE, FALSE
v ← TRUE
print(class(v))
o/p:- "logical"

ii) Numeric:- 12.3, 5, 999
v ← 23.5
print(class(v))
o/p:- "Numeric"

iii) Integer:- 2L, 3L, 0L
v ← 2L
print(class(v))
o/p:- "Integer"

iv) Complex 3+2i
v ← 2+5i
print(class(v))
o/p:- "Complex"

v) character "a", "good", '23.4'

```
v <- "TRUE"
print(class(v))
```

o/p:- "character"

(3) Program to create Pie chart:-

```
# create data for the graph
x <- c(21, 62, 10, 53)
labels <- c("London", "NYC", "Singapore", "Mumbai")

# Give the Pie chart file a name
png (file = "city.jpg")

# plot the chart
pie (x, labels)

# save the file
dev.off()
```

Program to create Bar chart:-

```
# create data for the chart
h <- c(7, 12, 28, 3, 41)

# Give the chart file a name
png (file = "barchart.jpg")

# plot the bar chart
barplot (H)

# save the file
dev.off
```

6. unlist()'-
This function will simplifies a list to produce a vector which contains all atomic component which occur in the list

Syntax'- unlist(x)
where x → is usually a list

eg'- test1 ← list(5, "b", 12)
unlist(list1)

o/p'- [1] "5" "b" "12"

ii) rbind()'- function combines vector, matrix or data frames by row

syntax'- rbind(my-data, new-row)

eg- x_1 ← c(7,4,4,9)
x_2 ← c(5,2,8,9)
x_3 ← c(1,2,3,4)

data-1 ← data.frame(x_1, x_2, x_3)

vector-1 ← c(9,8,7)

rbind(data-1, vector-1)

o/p'-

x_1	x_2	x_3
7	5	1
4	2	2
4	8	3
9	9	4
9	8	7

iii) cbind():- function combines vectors, matrix or data frames by columns.

cbind(x_1, x_2)
where x_1, x_2 can matrix, dataframes or vectors.

ex:- df1 = data.frame (
name = c("Rahul", "Joe", "Adam", "Ankit")
married_year = c(2016, 2015, 2015, 2012)
)

df2 = data.frame (
Birth_place = c("delhi", "NYC", "Tokyo", "del")
Birth_year = c(1997, 1990, 1992, 1989)
)

cbind (df1, df2)

o/p:-

name	married_year	Birth_place	Birth_year
Rahul	2016	delhi	1997
Joe	2015	NYC	1990
Adam	2015	Tokyo	1992
Ankit	2012	del	1989

iv) format():-
This function allows us to format an R object for pretty printing. It treats the elements of a vector as character strings using a common format.
format(): Pads string with same spaces so that they all have same length

Some arguments are:-

- width :- the minimum width of the string produced.
- trim :- if set to TRUE there is no Padding with spaces.
- justify :- controls how padding takes place for strings.

eg:-
format (c("A", "BB", "CCC"), width=5, justify = "centre")

o/p:- [1] "A" "BB" "CCC"

v) Barplot() :-
barplot can be created in R using barplot() function

Syntax:- barplot(x)

eg:-
max. temp ← c(22, 27, 29, 40, 37, 32)
barplot(max. temp).