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.

```
name <- "ramya"
> age <- 19
> name
[1] "ramya"
> age
[1] 19
> name = "r program"
```

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

```
> print("details of the object in memory :")
[1] "details of the object in memory :"
> print(ls.str())
age :  num 19
c :  cplx [1:5] 0+0i 0+0i 0+0i ...
chr :  chr [1:5] "" "" "" "" ""
l :  int 6
list : function (...)
listt : List of 3
 $ g1: int [1:9] 1 2 4 5 6 7 8 9 10
 $ g2: chr "r program"
 $ g3: chr "html"
mons_v :  chr [1:27] "March" "April" "January" ...
n1 :  num 10
n2 :  num 0.5
name :  chr "r program"
num :  num [1:6] 10 20 30 40 50 60
vec :  num [1:4] 1 2 3 4
```

x : num [1:6] 23 45 3 4 56 7

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.

```
> print("sequence of number from 20 to 50")
[1] "sequence of number from 20 to 50"
> print(seq(20,50))
 [1] 20 21 22 23 24 25 26 27 28 29 30 31 32
[14] 33 34 35 36 37 38 39 40 41 42 43 44 45
[27] 46 47 48 49 50
> print("mean of number from 20 to 60 :")
[1] "mean of number from 20 to 60 :"
> print(mean(20:60))
[1] 40
> print("sum of the numbers from 51 to 91")
[1] "sum of the numbers from 51 to 91"
> print(sum(51:91))
[1] 2911
```

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

```
 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] -22  38 -13  17  25 -11 -21  31  14 -40
```

5. Write a R program to get all prime numbers up to a given number (based on

the sieve of Eratosthenes).

```
> (prime_numbers <- function(n)
> { if (n >= 2) { x=seq(2,n) prime_nums = c() for (i in seq(2,n)) if (any(x == i)) { prime_nums = c(prime_nums, i) x = c(x %% i) !=0] ,i)}} return(prime_nums)}else{stop("input number should be atleast 2")}}prime_numbers(12)
[1]  2  3  5  7 11
```

6. Write a R program to extract first 10 english letter in lower case and last 10

letters in upper case and extract letters between 22 nd to 24 th letters in upper case.

```
print("First 10 letters in lower case:")
[1] "First 10 letters in lower case:"
t = head(letters, 10)
print(t)
[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
print("Last 10 letters in upper case:")
"Last 10 letters in upper case:"
t = tail(LETTERS, 10)
print(t)
print("Letters between 22nd to 24th letters in upper case:")
e = tail(LETTERS[22:24])
print(e)
 [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"
```

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

vector.

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

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

```
str1 = "The quick brown fox jumps over the lazy dog."
print("Original vector(string)")
[1] "Original vector(string)"
print(str1)
[1] "The quick brown fox jumps over the lazy dog."
print("Unique elements of the said vector:")
[1] "Unique elements of the said vector:"
print(unique(tolower(str1)))
[1] "the quick brown fox jumps over the lazy dog."
nums = c(1, 2, 2, 3, 4, 4, 5, 6)
print("Original vector(number)")
[1] "Original vector(number)"
print(nums)
[1] 1 2 2 3 4 4 5 6
print("Unique elements of the said vector:")
[1] "Unique elements of the said vector:"
print(unique(nums))
[1] 1 2 3 4 5 6
```

9. 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.

```
a<-c(1,2,3)
b<-c(4,5,6)
c<-c(7,8,9)
m<-cbind(a,b,c)
print("Content of the said matrix:")
[1] "Content of the said matrix:"
print(m)

     a b c
[1,] 1 4 7
[2,] 2 5 8
[3,] 3 6 9
```

10. Write a R program to create a list of random numbers in normal distribution and count occurrences of each value.

```
n = floor(rnorm(1000, 50, 100))
print('List of random numbers in normal distribution:')
[1] "List of random numbers in normal distribution:"
print(n)
  [1]   70   -5   88 -110  174  -66   10    8  106   35  -48  -54   12  102
 [15]   67  -34   83  -75   71   35   24  109   -3    9  221  231  105  105
 [29]  -58   43  -39   65   40   78  181   57   60  103  262  232  -77   31
 [43]   47  -50  174   67   41   92  243  -88   11  197 -117  126  -99   82
 [57]  -76  245   27  -35  -79  -14 -278   -6   -3   79  229  -15  -11   52
 [71]  127  -14  150   42  184  -45  -37   87   11  146  124  158  113  166
```

[85] -11 47 -23 -99 63 -7 0 -2 30 -146 141 29 73 114

[99] -16 -14 116 -80 -29 22 352 -60 126 12 287 23 -7 149

[113] 11 40 65 239 94 119 76 47 159 -75 103 91 88 51

[127] 115 51 -170 224 144 133 272 -229 125 -32 39 8 62 105

[141] 21 -70 51 31 223 31 -71 105 -73 -48 -50 117 208 229

[155] 97 134 113 76 -137 -17 -160 -8 -30 53 161 -90 155 -9

[169] 118 -83 101 91 201 146 18 -4 77 -76 142 141 43 95

[183] -56 55 -43 -151 68 51 180 142 53 -27 59 99 39 49

[197] 4 14 201 55 159 -141 161 -113 -46 108 139 143 104 7

[211] -34 21 290 -53 117 245 -44 75 -89 -45 -19 -75 64 205

[225] 27 22 18 99 -13 -25 33 16 281 -169 224 207 124 167

[239] 113 81 42 74 59 -40 155 -40 112 -27 3 60 2 82

[253] 158 78 117 -55 172 6 149 209 -52 -88 62 158 71 34

[267] 146 179 22 92 187 178 -3 119 -25 165 -27 10 58 -25

[281] -37 155 -47 27 -44 -57 57 8 -89 142 52 -19 -116 -80

[295] -55 71 239 187 -55 53 -115 -132 95 21 -48 34 72 -8

[309] 71 70 93 -190 192 189 -72 -19 37 164 -91 30 -109 -79

[323] 11 164 277 -89 156 279 -25 11 78 27 2 16 -34 51

[337] 21 -124 51 -107 51 19 331 77 2 144 103 -13 77 -52

[351] -9 33 54 56 108 204 -24 -5 109 -20 35 -25 61 -20

[365] -33 172 -98 39 131 -72 2 47 168 46 -8 215 -126 154

[379] 167 17 128 185 95 12 -14 -49 -78 246 11 15 -214 -160

[393] 265 -134 61 -54 -38 -26 156 25 41 108 114 72 -45 -29

[407] 178 55 65 88 44 1 -141 136 152 50 74 229 132 -3

[421] 70 -7 115 101 -53 -42 75 -159 8 118 -48 115 38 -17

[435] 121 -170 189 126 53 -10 119 -50 98 -20 161 -48 24 -54

[449] -47 -101 121 33 179 319 26 80 -61 -77 19 186 129 185

[463] 338 -46 102 -303 98 230 109 -74 -37 78 -58 212 82 57

[477] 186 154 76 6 -111 -119 -23 24 25 148 -39 -49 -36 175

[491] 139 46 138 5 -113 86 -89 301 118 -66 102 207 57 165

[505] 18 63 105 67 87 -90 -77 -81 197 286 -169 22 28 60

[519] 58 27 47 49 162 87 -88 189 -63 57 126 -30 70 83

[533] -79 -160 -110 3 -17 141 164 60 -54 24 -96 120 242 -17

[547] 181 -64 147 47 3 62 -195 -148 246 145 98 -50 -42 -24

[561] 94 159 10 211 -129 115 -111 127 -9 -80 -83 108 121 13

[575] 13 -120 20 46 -91 41 -54 36 -39 245 -6 7 264 67

[589] -18 13 0 -76 195 125 102 -43 23 97 -107 89 49 56

[603] 65 -98 94 111 241 -29 262 17 80 50 -32 100 182 201

[617] 115 259 52 181 121 93 192 126 96 -126 263 130 -194 14

[631] 16 173 87 -39 172 126 160 91 185 192 99 152 3 -107

[645] 46 -88 98 -80 98 209 -129 58 16 206 76 119 116 329

[659] 162 -14 195 -13 238 199 208 -82 99 62 74 165 187 111

[673] 3 60 60 -19 -35 176 -94 4 244 47 130 -2 34 175

[687] 53 14 14 29 171 136 -21 217 87 117 80 -4 -23 197

[701] 113 267 4 -76 12 7 42 242 63 61 41 -29 77 243

[715] 124 -78 -43 58 67 73 3 70 97 49 140 -104 166 -34

[729] -58 142 -50 113 46 126 119 60 28 223 64 164 71 -76

[743] 93 21 -23 166 6 26 -4 66 83 61 75 45 134 23

[757] 62 117 -14 218 36 41 -73 42 196 53 52 -21 19 -74

[771] -31 47 45 -36 53 -63 173 41 133 -77 222 200 158 202

[785] -4 272 101 192 167 74 133 85 -84 -42 108 147 -82 46

[799] -90 111 163 -74 20 22 1 -73 238 85 112 41 36 52

[813] 39 2 21 -104 155 -161 199 -149 20 109 111 81 -137 53

[827] 212 -51 -15 116 -3 119 71 110 -35 98 -88 0 -34 93

[841] 39 -65 125 13 142 -29 -153 -32 -19 22 176 -248 136 155

[855] -10 15 30 17 76 214 170 61 -98 108 -7 100 37 93

[869] 1 -12 -35 154 200 100 -4 168 162 6 198 164 -45 -56

[883] 106 84 162 163 66 -9 93 43 195 56 -16 76 57 236

[897] 85 106 -72 70 -170 166 -57 110 -17 77 314 -225 134 -140

[911] 95 134 -3 67 236 -23 162 -13 47 232 -74 -15 164 -100

[925] 80 20 -72 33 192 159 87 170 95 -33 154 -21 139 156

[939] -74 138 136 41 73 88 253 129 -12 24 22 -13 15 52

```
 [953]  58 -39  85  67 -28   7  38 -40 170  95  -4  89 131  55
 [967]  78  60 -43  81 -54 -78  32  33 -36 215 -64  58  18 -87
 [981] 163  74 288 -44  17 -122 143 116 192  29 144   6 180 -124
 [995] 167 -62  53 -30  62 158
t = table(n)
print("Count occurrences of each value:")
[1] "Count occurrences of each value:"
print(t)
n
-303 -278 -248 -229 -225 -214 -195 -194 -190 -170 -169 -161 -160 -159 -153 -151
   1    1    1    1    1    1    1    1    1    3    2    1    3    1    1    1
-149 -148 -146 -141 -140 -137 -134 -132 -129 -126 -124 -122 -120 -119 -117 -116
   1    1    1    2    1    2    1    1    2    2    2    1    1    1    1    1
-115 -113 -111 -110 -109 -107 -104 -101 -100  -99  -98  -96  -94  -91  -90  -89
   1    2    2    2    1    3    2    1    1    2    3    1    1    2    3    4
 -88  -87  -84  -83  -82  -81  -80  -79  -78  -77  -76  -75  -74  -73  -72  -71
   5    1    1    2    2    1    4    3    3    4    5    3    5    3    4    1
 -70  -66  -65  -64  -63  -62  -61  -60  -58  -57  -56  -55  -54  -53  -52  -51
   1    2    1    2    2    1    1    1    3    2    2    3    6    2    2    1
 -50  -49  -48  -47  -46  -45  -44  -43  -42  -40  -39  -38  -37  -36  -35  -34
   5    2    5    2    2    4    3    4    3    3    5    1    3    3    4    5
 -33  -32  -31  -30  -29  -28  -27  -26  -25  -24  -23  -21  -20  -19  -18  -17
   2    3    1    3    5    1    3    1    5    2    5    3    3    5    1    5
 -16  -15  -14  -13  -12  -11  -10   -9   -8   -7   -6   -5   -4   -3   -2    0
   2    3    6    5    2    2    2    4    3    4    2    2    6    6    2    3
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
   3    5    6    3    1    5    4    4    1    3    6    4    4    4    3    4
  17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
   4    4    3    4    6    7    3    5    2    2    5    2    3    3    3    1
  33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   49
   5    3    3    3    2    2    5    2    8    4    3    1    2    6    9    4
```

```
 50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65
  2   7   6   9   1   4   3   6   6   2   8   5   6   3   2   4
 66  67  68  70  71  72  73  74  75  76  77  78  79  80  81  82
  2   7   1   6   6   2   3   5   3   6   5   5   1   4   3   3
 83  84  85  86  87  88  89  91  92  93  94  95  96  97  98  99
  3   1   4   1   6   4   2   3   2   6   3   6   1   3   6   4
100 101 102 103 104 105 106 108 109 110 111 112 113 114 115 116
  3   3   4   3   1   5   3   6   4   2   4   2   5   2   5   4
117 118 119 120 121 124 125 126 127 128 129 130 131 132 133 134
  5   3   6   1   4   3   3   7   2   1   2   2   2   1   3   4
136 138 139 140 141 142 143 144 145 146 147 148 149 150 152 154
  4   2   3   1   3   5   2   3   1   3   2   1   2   1   2   4
155 156 158 159 160 161 162 163 164 165 166 167 168 170 171 172
  5   3   5   4   1   3   5   3   6   3   4   4   2   3   1   3
173 174 175 176 178 179 180 181 182 184 185 186 187 189 192 195
  2   2   2   2   2   2   2   3   1   1   3   2   3   3   6   3
196 197 198 199 200 201 202 204 205 206 207 208 209 211 212 214
  1   3   1   2   2   3   1   1   1   1   2   2   2   1   2   1
215 217 218 221 222 223 224 229 230 231 232 236 238 239 241 242
  2   1   1   1   1   2   2   3   1   1   2   2   2   2   1   2
243 244 245 246 253 259 262 263 264 265 267 272 277 279 281 286
  2   1   3   2   1   1   2   1   1   1   1   2   1   1   1   1
287 288 290 301 314 319 329 331 338 352
  1   1   1   1   1   1   1   1   1   1
```

11. Write a R program to create three vectors numeric data, character data and logical data. Display the content of the vectors and their type.

```
a = c(1, 2, 5, 3, 4, 0, -1, -3)
b = c("Red", "Green", "White")
c = c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)
```

```
print(a)
[1]  1  2  5  3  4  0 -1 -3
print(typeof(a))
[1] "double"
print(b)
[1] "Red"   "Green" "White"
print(typeof(b))
[1] "character"
print(c)
[1]  TRUE  TRUE  TRUE FALSE  TRUE FALSE
print(typeof(c))
"logical"
```

12. 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.

```
m1 = matrix(1:20, nrow=5, ncol=4)
print("5 × 4 matrix:")
print(m1)
cells = c(1,3,5,7,8,9,11,12,14)
rnames = c("Row1", "Row2", "Row3")
cnames = c("Col1", "Col2", "Col3")
m2 = matrix(cells, nrow=3, ncol=3, byrow=TRUE, dimnames=list(rnames, cnames))
print("3 × 3 matrix with labels, filled by rows: ")
print(m2)
print("3 × 3 matrix with labels, filled by columns: ")
m3 = matrix(cells, nrow=3, ncol=3, byrow=FALSE, dimnames=list(rnames, cnames))
print(m3)
[1] "5 × 4 matrix:"
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
```

```
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
[1] "3 × 3 matrix with labels, filled by rows: "
     Col1 Col2 Col3
Row1    1    3    5
Row2    7    8    9
Row3   11   12   14
[1] "3 × 3 matrix with labels, filled by columns: "
     Col1 Col2 Col3
Row1    1    7   11
Row2    3    8   12
Row3    5    9   14
```

13. 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.

```
a =  array(
  6:30,
  dim = c(4, 3, 2),
  dimnames = list(
    c("Col1", "Col2", "Col3", "Col4"),
    c("Row1", "Row2", "Row3"),
    c("Part1", "Part2")
  )
)
print(a)
, , Part1

     Row1 Row2 Row3
```

Col1 6 10 14

Col2 7 11 15

Col3 8 12 16

Col4 9 13 17

, , Part2

Row1 Row2 Row3

Col1 18 22 26

Col2 19 23 27

Col3 20 24 28

Col4 21 25 29

14. 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.

```
v1 = c(1, 3, 5, 7)
v2 = c(2, 4, 6, 8, 10)
arra1 = array(c(v1, v2),dim = c(3,3,2))
print(arra1)
```

, , 1

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

[1,] 1 7 6

[2,] 3 2 8

[3,] 5 4 10

, , 2

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

[1,] 1 7 6

[2,] 3 2 8

[3,] 5 4 10

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

```
l = list(
  c(1, 2, 2, 5, 7, 12),
  month.abb,
  matrix(c(3, -8, 1, -3), nrow = 2),
  asin
)
print("Content of the list:")
print(l)
```

[1] "Content of the list:"

[[1]]

[1] 1 2 2 5 7 12

[[2]]

 [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"

[[3]]

     [,1] [,2]

[1,] 3 1

[2,] -8 -3

[[4]]

function (x) .Primitive("asin")