In [1]:

```
a=readline("Enter the number: ")
```

```
Enter the number: 15
```

In [2]:

```
class(a)
```

'character'

Converting the value while getting the value from the user:

In [3]:

```
b=as.integer(readline("Enter the number:"))
```

```
Enter the number:453
```

In [4]:

```
class(b)
```

'integer'

In [5]:

```
d=c(1,2,3,4,5,6,7,8,9,0)
```

In [6]:

```
d
```

```
1  2  3  4  5  6  7  8  9  0
```

In [7]:

```
d[2]
```

2

In [8]:

```
# accessing the values in the array
d[c(1,3)]
```

```
1  3
```

In [9]:

```
# start : stop
d[1:5]
```

```
1   2   3   4   5
```

In [10]:

```
#updating the values in the vectors:
d[c(2,5)]=100
```

In [11]:

```
d
```

```
1   100   3   4   100   6   7   8   9   0
```

In [12]:

```
# sorting the values in ascending
sort(d)
```

```
0   1   3   4   6   7   8   9   100   100
```

In [13]:

```
# sorting the values in descending
d=sort(d,TRUE)
```

In [14]:

```
d
```

```
100   100   9   8   7   6   4   3   1   0
```

In [15]:

```
# ceiling => for next number
# floor => for the previous number

ceiling(10.5)
floor(10.5)
```

```
11

10
```

In [16]:

```
d=rep(c(1,2,3),times=4)
d
```

```
1   2   3   1   2   3   1   2   3   1   2   3
```

In [17]:

```
e=c(1:100)
e
```

```
1   2   3   4   5   6   7   8   9   10   11   12   13   14   15   16   17   18   19   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  51  52  53  54  55  56  57  58  59  60  61  62
63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82
83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100
```

SEQUENCE:

BY --> TO JUMP THE VALUE

LENGTH.OUT --> VALUE COUNT BETWEEN THE RANGE

In [18]:

```
# by sequence:
seq(from=10, to=100, by=2)
```

```
10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40  42  44  46  48
50  52  54  56  58  60  62  64  66  68  70  72  74  76  78  80  82  84  86  88
90  92  94  96  98  100
```

In [19]:

```
# length.out sequence:
seq(from=1,to=5, length.out=25)
```

```
1  1.16666666666667  1.33333333333333  1.5  1.66666666666667  1.83333333333333
2  2.16666666666667  2.33333333333333  2.5  2.66666666666667  2.83333333333333
3  3.16666666666667  3.33333333333333  3.5  3.66666666666667  3.83333333333333
4  4.16666666666667  4.33333333333333  4.5  4.66666666666667  4.83333333333333
5
```

In [ ]:

```

```

In [ ]:

```

```

In [20]:

```
mark=c(1,2,3,4,5,6,7,8,9,5,4,6,7,2,4)
```

In [21]:

```
mark
```

```
1  2  3  4  5  6  7  8  9  5  4  6  7  2  4
```

In [22]:

```
print(paste("The Mark 2 count is: ", sum(mark==2)))
```

```
[1] "The Mark 2 count is:  2"
```

In [23]:

```
sub=c("java","python", "r", "Data", "python")
sub
```

```
'java'  'python'  'r'  'Data'  'python'
```

In [24]:

```
a=readline("Enter your favourite language:")
```

```
Enter your favourite language:python
```

In [25]:

```
print(paste("Your favourite language repeated",sum(sub==a),"times"))
```

```
[1] "Your favourite language repeated 2 times"
```

In [ ]:

```
UNIQUE VALUES:
```

In [26]:

```
mark
```

```
1  2  3  4  5  6  7  8  9  5  4  6  7  2  4
```

In [27]:

```
unique(mark)
```

```
1  2  3  4  5  6  7  8  9
```

In [ ]:

```
                                        ★————————————————★
```

In [29]:

```
# SET DIFFERENCE VALUES A-B
AIMLmark=c(10,22,34,45,6,34,53,89)
CYBERmark=c(15,60,45,36,48,48,21,24)
```

In [30]:

```
setdiff(AIMLmark,CYBERmark)
```

```
10  22  34  6  53  89
```

In [31]:

```
setdiff(CYBERmark,AIMLmark)
```

```
15  60  36  48  21  24
```

In [ ]:

```
                                   ★————————————★
```

GETTING MULTIPLE VALUES FROM THE USER (OR) TO EXECUTE THE VALUES MANY NUMBER OF TIMES

In [33]:

```
element=c()
total=as.integer(readline("Enter the number: "))
for(i in seq(total))
    {
       element[i]=as.numeric(readline(paste("Enter the element",i,":")))
}
```

```
Enter the number: 5
Enter the element 1 :45
Enter the element 2 :65
Enter the element 3 :78
Enter the element 4 :25
Enter the element 5 :15
```

In [34]:

```
element
```

```
45  65  78  25  15
```

In [35]:

```
sort(element)
```

```
15  25  45  65  78
```

In [36]:

```
sort(element,TRUE)
```

```
78   65   45   25   15
```

```
                                                   ★————————————————————★
```

ARRAY --> n dimentional

In [37]:

```
array(c(1,2,3,4,5,6,7,8,9), dim=c(3,3))
```

```
1  4  7
2  5  8
3  6  9
```

Create a array 2 (3x3) matrix each with 3 rows and 3 columns using two given vectors:

In [38]:

```
v1=c(1,2,3,4)
v2=c(45,56,87,29,34,459,5667,459)
```

In [39]:

```
mat=array(c(v1,v2), dim=c(3,3,2))
print(mat)
```

```
, , 1

     [,1] [,2] [,3]
[1,]    1    4   87
[2,]    2   45   29
[3,]    3   56   34

, , 2

     [,1] [,2] [,3]
[1,]  459    1    4
[2,] 5667    2   45
[3,]  459    3   56
```

For creating 3 matrix with the dimension 3x3:

In [41]:

```
mat=array(c(v1,v2), dim=c(3,3,3))
print(mat)
```

```
, , 1

     [,1] [,2] [,3]
[1,]    1    4   87
[2,]    2   45   29
[3,]    3   56   34

, , 2

     [,1] [,2] [,3]
[1,]  459    1    4
[2,] 5667    2   45
[3,]  459    3   56

, , 3

     [,1] [,2] [,3]
[1,]   87  459    1
[2,]   29 5667    2
[3,]   34  459    3
```

Print the second row of the second matrix of an array

In [42]:

```
mat[2,,2] # mat[row, column, matrix no.]
```

```
5667  2  45
```

Print the third row of third column of first matrix

In [45]:

```
mat[3,3,1]
```

```
34
```

Write a R program to create a 2 dimensional 5x3 array of sequence of even integers greater than 50

In [46]:

```
array(seq(from=50,length.out=15,by=2),dim=c(5,3))
```

```
50  60  70
52  62  72
54  64  74
56  66  76
58  68  78
```

In [47]:

```
# ROUGH:
array(seq(from=50,length.out=15,by=2),dim=c(3,5))
```

```
50  56  62  68  74
52  58  64  70  76
54  60  66  72  78
```

In [ ]:

```
                          ★————————————★
```

MATRIX:

In [ ]:

```
# syntax
# matrix(c(), nrow=, ncol=)
```

In [51]:

```
matrix(c(1:16), nrow=4, ncol=4)
```

```
1  5   9  13
2  6  10  14
3  7  11  15
4  8  12  16
```

In [52]:

```
# byrow function is used for passing the values by rows instead of column wise:
matrix(c(1:16), nrow=4, ncol=4, byrow=TRUE)
```

```
 1   2   3   4
 5   6   7   8
 9  10  11  12
13  14  15  16
```

Matrix with the given values

In [59]:

```
rname=c("r1","r2","r3","r4")
cname=c("c1","c2","c3","c4")
a=matrix(c(1:16), nrow=4, ncol=4, byrow=TRUE, dimnames=list(rname,cname))
a
```

	c1	c2	c3	c4
r1	1	2	3	4
r2	5	6	7	8
r3	9	10	11	12
r4	13	14	15	16

In [60]:

```
# accessing the element in the matrix:
a[2]
```

```
5
```

In [61]:

```
a[2,2]
```

```
6
```

In [62]:

```
#displaying second column
a[,2]
```

```
r1
2
r2
6
r3
10
r4
14
```

In [63]:

```
# displaying second row
a[2,]
```

```
c1
5
c2
6
c3
7
c4
8
```

In [ ]:

```
★━━━━━━━━━━━━━━━━★
```

FACTOR:

It stores the categorical data

In [64]:

```
a=factor(c("Java","Python","Java", "Python", "R", "C"))
```

In [65]:

```
a
```

```
Java   Python   Java   Python   R   C
```

▼ **Levels**:

```
'C'   'Java'   'Python'   'R'
```

In [66]:

```
table(a) #gives the count of each value in a tabular form
```

```
a
     C   Java Python      R
     1      2      2      1
```

In [67]:

```
levels(a)
```

'C' 'Java' 'Python' 'R'

Get input from user for list:

In [70]:

```
element=list()
total=as.integer(readline("Enter the number: "))
for(i in seq(total))
    {
       element[i]=as.numeric(readline(paste("Enter the element",i,":")))
}
```

```
Enter the number: 5
Enter the element 1 :12
Enter the element 2 :45
Enter the element 3 :78
Enter the element 4 :6
Enter the element 5 :54
```

In [71]:

```
element
```

1. 12
2. 45
3. 78
4. 6
5. 54

DATAFRAME:

--> Table Structure

--> Each Column should contain same no. of data table items

In [83]:

```
df=data.frame(emp_id=c(1,2,3),
              emp_name=c("Nanthiesh","Demon","NarutoUzumaki"),
              emp_date=c("2004-07-27", "1001-01-01", "1001-03-01"),
              gender=factor(c("M","M","F")))
```

In [84]:

```
df
```

emp_id	emp_name	emp_date	gender
1	Nanthiesh	2004-07-27	M
2	Demon	1001-01-01	M
3	NarutoUzumaki	1001-03-01	F

In [85]:

```
summary(df)
```

```
     emp_id                emp_name        emp_date gender
 Min.   :1.0   Demon        :1   1001-01-01:1   F:1
 1st Qu.:1.5   Nanthiesh    :1   1001-03-01:1   M:2
 Median :2.0   NarutoUzumaki:1   2004-07-27:1
 Mean   :2.0
 3rd Qu.:2.5
 Max.   :3.0
```

In [86]:

```
str(df)
```

```
'data.frame':	3 obs. of  4 variables:
 $ emp_id  : num  1 2 3
 $ emp_name: Factor w/ 3 levels "Demon","Nanthiesh",..: 2 1 3
 $ emp_date: Factor w/ 3 levels "1001-01-01","1001-03-01",..: 3 1 2
 $ gender  : Factor w/ 2 levels "F","M": 2 2 1
```

In [87]:

```
df[,2]
```

Nanthiesh Demon NarutoUzumaki

▼ **Levels**:

'Demon' 'Nanthiesh' 'NarutoUzumaki'

In [89]:

```
df[3,]
```

	emp_id	emp_name	emp_date	gender
3	3	NarutoUzumaki	1001-03-01	F

In [92]:

```
table(df$gen)
```

```
F M
1 2
```

In [93]:

```
df["emp_name"]
```

emp_name
Nanthiesh
Demon
NarutoUzumaki

In [94]:

```
df$emp_name
```

Nanthiesh Demon NarutoUzumaki

▼ **Levels**:

'Demon' 'Nanthiesh' 'NarutoUzumaki'

In [95]:

```
# to find only the employee name and employee date

data.frame(df$emp_name, df$emp_date)
```

df.emp_name	df.emp_date
Nanthiesh	2004-07-27
Demon	1001-01-01
NarutoUzumaki	1001-03-01

In [99]:

```
# sorting the data with joining date

df[with(df,order(c(emp_date)))]
```

emp_name	emp_date	emp_id
Nanthiesh	2004-07-27	1
Demon	1001-01-01	2
NarutoUzumaki	1001-03-01	3

In [97]:

```
# sorting the data with employee name

df[with(df,order(c(emp_name)))]
```

emp_name	emp_id	emp_date
Nanthiesh	1	2004-07-27
Demon	2	1001-01-01
NarutoUzumaki	3	1001-03-01