1.1)Write a R program to create two 2x3 matrix and add, subtract, multiply and divide the matrixes.

```
# Create two 2x3 matrixes.
m1 = matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
print("Matrix-1:")
print(m1)
m2 = matrix(c(0, 1, 2, 3, 0, 2), nrow = 2)
print("Matrix-2:")
print(m2)

result = m1 + m2
print("Result of addition")
print(result)

result = m1 - m2
print("Result of subtraction")

print(result)

result = m1 * m2
print("Result of multiplication")
print(result)

result = m1 / m2
print("Result of division:")
print(result)
```

1.2)Write a R program to create an array of two 3x3 matrices each with 3 rows and 3 columns from two given two vectors.

```
print("Two vectors of different lengths:")
v1 =  c(1,3,4,5)
v2 =  c(10,11,12,13,14,15)
print(v1)
print(v2)
result = array(c(v1,v2),dim = c(3,3,2))
print("New array:")
print(result)
```

1.3)Write a R program to combine three arrays so that the first row of the first array is followed by the first row of the second array and then first row of the third array.

```
num1 = rbind(rep("A",3), rep("B",3), rep("C",3))
print("num1")
print(num1)
num2 = rbind(rep("P",3), rep("Q",3), rep("R",3))
print("num2")
print(num2)
num3 = rbind(rep("X",3), rep("Y",3), rep("Z",3))
print("num3")
print(num3)
a = matrix(t(cbind(num1,num2,num3)),ncol=3, byrow=T)
print("Combine three arrays, taking one row from each one by one:")
print(a)
```

1.4)Write a R program to convert a given matrix to a 1 dimensional array.

```
m=matrix(1:12,3,4)
print("Original matrix:")
print(m)
a = as.vector(m)
print("1 dimensional array:")
print(a)
```

2.1)Write a R program to create a data frame from four given vectors.

```
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily',
'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas')
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1)
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes')
print("Original data frame:")
print(name)
print(score)
print(attempts)
print(qualify)
df = data.frame(name, score, attempts, qualify)
print(df)
```

2.2)Write a R program to get the structure of a given data frame.

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily',
'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes')
)
print("Original dataframe:")
print(exam_data)
print("Structure of the said data frame:")
print(str(exam_data))
```

2.3)Write a R program to extract specific column from a data frame using column name.

```
exam_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily',
'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes')

)

print("Original dataframe:")

print(exam_data)

print("Extract Specific columns:")

result <- data.frame(exam_data$name,exam_data$score)

print(result)
```

2.4)Write a R program to add new row(s) to an existing data frame.

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily',
'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes')
)
print("Original dataframe:")
print(exam_data)
new_exam_data = data.frame(
name = c('Robert', 'Sophia'),
score = c(10.5, 9),
attempts = c(1, 3),
qualify = c('yes', 'no')
)

exam_data =  rbind(exam_data, new_exam_data)
print("After adding new row(s) to an existing data frame:")
print(exam_data)
```

## 2.5) Write a R program to add a new column in a given data frame.

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily',
'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes')
)
print("Original dataframe:")
print(exam_data)
print("New data frame after adding the 'country' column:")
exam_data$country =
c("USA","USA","USA","USA","USA","USA","USA","USA","USA","USA")
print(exam_data)
```

## 2.6) Write a R program to drop column(s) by name from a given data frame.

```
exam_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
print("Original dataframe:")
print(exam_data)
exam_data = subset(exam_data, select = -c(name, qualify))
print(exam_data)
```

## 2.7)Write a R program to sort a given data frame by multiple column(s).

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
print("Original dataframe:")
print(exam_data)
print("dataframe after sorting 'name' and 'score' columns:")
exam_data = exam_data[with(exam_data, order(name, score)), ]
print(exam_data)
```

## 2.8) Write a R program to replace NA values with 3 in a given data frame.

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
```

```
print("Original dataframe:")
print(exam_data)
exam_data[is.na(exam_data)] = 3
print("After removing NA with 3, the said dataframe becomes:")
print(exam_data)
```

## 2.9) Write a R program to count the number of NA values in a data frame column.

```
exam_data = data.frame(
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
print("Original dataframe:")
print(exam_data)
print("The number of NA values in attempts column:")
print(sum(is.na(exam_data$attempts)))
```

## 2.10)Write a R program to find elements which are present in two given data frames.

```
a = c("a", "b", "c", "d", "e")
b = c("d", "e", "f", "g")
print("Original Dataframes")
print(a)
print(b)
print("Elements which are present in both dataframe:")
```

```
result = intersect(a, b)
print(result)
```

### 3.1)Write a R program to create a vector of a specified type and length. Create vector of numeric, complex, logical and character types of length 6.

```
x = vector("numeric", 5)
print("Numeric Type:")
print(x)
c = vector("complex", 5)
print("Complex Type:")
print(c)
l = vector("logical", 5)
print("Logical Type:")
print(l)
chr = vector("character", 5)
print("Character Type:")
print(chr)
```

### 3.2)Write a R program to add two vectors of integers type and length 3.

```
x = c(10, 20, 30)
 y = c(20, 10, 40)
 print("Original Vectors:")
 print(x)
 print(y)
 print("After adding two Vectors:")
 z = x + y
 print(z)
```

### 3.3)Write a R program to append value to a given empty vector.

```
vector = c()

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

for (i in 1:length(values))

  vector[i] <- values[i]

print(vector)
```

### 3.4)Write a R program to multiply two vectors of integers type and length 3.

```
x = c(10, 20, 30)

 y = c(20, 10, 40)

 print("Original Vectors:")

 print(x)

 print(y)

 print("Product of two Vectors:")

 z = x * y

 print(z)
```

### 3.5)Write a R program to find Sum, Mean and Product of a Vector.

```
x = c(10, 20, 30)

print("Sum:")

print(sum(x))

print("Mean:")

print(mean(x))

print("Product:")

print(prod(x))
```

## 3.6)Write a R program to find Sum, Mean and Product of a Vector, ignore element like NA or NaN.

```
x = c(10, NULL, 20, 30, NA)
print("Sum:")
#ignore NA and NaN values
print(sum(x, na.rm=TRUE))
print("Mean:")
print(mean(x, na.rm=TRUE))
print("Product:")
print(prod(x, na.rm=TRUE))
```

## 3.7)Write a R program to find the minimum and the maximum of a Vector.

```
x = c(10, 20, 30, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Maximum value of the above Vector:")
print(max(x))
print("Minimum value of the above Vector:")
print(min(x))
```

## 3.8)Write a R program to sort a Vector in ascending and descending order.

```
x = c(10, 20, 30, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Sort in ascending order:")
print(sort(x))
print("Sort in descending order:")
print(sort(x, decreasing=TRUE))
```

## 3.8) Write a R program to count the specific value in a given vector.

```
x = c(10, 20, 30, 20, 20, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Count specific value(20) in above vector:")
print(sum(x==20))
```

## 3.9)Write a R program to access the last value in a given vector.

```
x = c(10, 20, 30, 20, 20, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Access the last value of the said vector:")
print(tail(x, n=1))
```

## 4.1)Write a R program to create a list containing a vector, a matrix and a list and remove the second element.

```
list_data <- list(c("Red","Green","Black"), matrix(c(1,3,5,7,9,11), nrow = 2),
list("Python", "PHP", "Java"))
print("List:")
print(list_data)
print("Remove the second element of the list:")
list_data[2] = NULL
print("New list:")
print(list_data)
```

### 4.2)Write a R program to create a list containing a vector, a matrix and a list and update the last element.

```
list_data <- list(c("Red","Green","Black"), matrix(c(1,3,5,7,9,11), nrow = 2),
list("Python"))
print("List:")
print(list_data)
print("Update the second element of the list:")
list_data[3] =  "R programming"
print("New list:")
print(list_data)
```

### 4.3)Write a R program to merge two given lists into one list.

```
n1 = list(1,2,3)
c1 = list("Red", "Green", "Black")
print("Original lists:")
print(n1)
print(c1)
print("Merge the said lists:")
mlist =  c(n1, c1)
print("New merged list:")
print(mlist)
```

### 5.1)Write a R program to create an ordered factor from data consisting of the names of months.

```
mons_v = c("March","April","January","November","January",
"September","October","September","November","August","February",
"January","November","November","February","May","August","February",
"July","December","August","August","September","November","September",
"February","April")
print("Original vector:")
print(mons_v)
f = factor(mons_v)
print("Ordered factors of the said vector:")
print(f)
print(table(f))
```

## 5.2)Write a R program to find the levels of factor of a given vector.

```
v = c(1, 2, 3, 3, 4, NA, 3, 2, 4, 5, NA, 5)
print("Original vector:")
print(v)
print("Levels of factor of the said vector:")
print(levels(factor(v)))
```

## 5.3)Write a R program to change the first level of a factor with another level of a given factor.

```
v = c("a", "b", "a", "c", "b")
print("Original vector:")
print(v)
f = factor(v)
print("Factor of the said vector:")
print(f)
levels(f)[1] = "e"
print(f)
```

## 5.4)Write a R program to create an ordered factor from data consisting of the names of months.

```
mons_v = c("March","April","January","November","January",

"September","October","September","November","August","February",

"January","November","November","February","May","August","February",

"July","December","August","August","September","November","September",

"February","April")

print("Original vector:")

print(mons_v)

f = factor(mons_v)

print("Ordered factors of the said vector:")

print(f)

print(table(f))
```

## 5.5)Write a R program to concatenate two given factor in a single factor.

```
f1 <- factor(sample(LETTERS, size=6, replace=TRUE))

f2 <- factor(sample(LETTERS, size=6, replace=TRUE))

print("Original factors:")

print(f1)

print(f2)

f = factor(c(levels(f1)[f1], levels(f2)[f2]))

print("After concatenate factor becomes:")

print(f)
```