R ASSIGNMENT

NAME: AYUSH NANDI

ROLL NO: 22MCF1R13

1. **Write R function to initialize a data frame of 10 people with columns as name and height (in cm). Use for loop to convert the height in to meter.**

```
# q1


initialize_df = function(x, y)
{
    df = data.frame(names=x, height=y)
}


to_meter = function(df)
{
    m = numeric(0)
    for(h in df$height){ m = c(m, h/100) }

    new_df = data.frame(names=df$names, height_in_m=m)
}


n = scan(what='')
h = scan()
people_cm = initialize_df(n, h)
people_cm
people_m = to_meter(people_cm)
people_m
```

OUTPUT

```
1    ayush    200
2   rishav    180
3     aman    160
4  koushil    165
5    depak    162
6   dipesh    178
7    joshi    172
8   puneet    177
9   aishik    180
10 prateek    169
> people_m = to_meter(people_cm)
> people_m
     names height_in_m
1    ayush        2.00
2   rishav        1.80
3     aman        1.60
4  koushil        1.65
5    depak        1.62
6   dipesh        1.78
7    joshi        1.72
8   puneet        1.77
9   aishik        1.80
10 prateek        1.69
```

2. The numbers of 10 days of rainfall amounts are given. Read them into a vector and calculate the mean and standard deviation, along with the days of highest and lowest rainfall.

```
# q2


rainfall_amt = scan()
print(paste("mean is ", mean(rainfall_amt)))
print(paste('standard deviation is ', sd(rainfall_amt)))


highest = max(rainfall_amt)
lowest = min(rainfall_amt)


for(i in 1:length(rainfall_amt))
{
    if(rainfall_amt[i] == highest){ print(paste("highest rainfall amount recieved in day", (i)))}
    if(rainfall_amt[i] == lowest){ print(paste("lowest rainfall amount recieved in day", (i)))}
}
```

OUTPUT

```
> print(paste("mean is ", mean(rainfall_amt)))
[1] "mean is  8.44"
> print(paste('standard deviation is ', sd(rainfall_amt)))
[1] "standard deviation is  13.6647315703196"
> for(i in 1:length(rainfall_amt))
+ {
+ if(rainfall_amt[i] == highest){ print(paste("highest rainfall amount recie$
+ if(rainfall_amt[i] == lowest){ print(paste("lowest rainfall amount recieve$
+ }
[1] "highest rainfall amount recieved in day 3"
[1] "lowest rainfall amount recieved in day 9"
```

3. Consider a matrix filled with 10 rows and 10 columns all filled with random numbers between 0 and 1. Calculate row means and column means, sum of all diagonal elements and standard deviation across both

```
# q3


mat = matrix(runif(100, 0, 1), 10, 10)
mat


# (i)
rmean = rowMeans(mat)
print('row means are')
print(rmean)
print('column means are')
cmean = colMeans(mat)
print(cmean)


# (ii)
sum = sum(diag(mat))
print(paste('sum of diagonals', sum))


# (iii)
print(paste('standard deviation is', sd(mat)))
```

OUTPUT

```
> print('row means are')
[1] "row means are"
> print(rmean)
 [1] 0.4477068 0.5470458 0.5280373 0.5680646 0.4660293 0.6661798 0.4029690
 [8] 0.4249001 0.4309709 0.5163341
> print('column means are')
[1] "column means are"
> cmean = colMeans(mat)
> print(cmean)
 [1] 0.5313650 0.5637694 0.4743984 0.5861029 0.3590758 0.3784927 0.5414057
 [8] 0.5499661 0.5080719 0.5055899
> sum = sum(diag(mat))
> print(paste('sum of diagonals', sum))
[1] "sum of diagonals 3.86040217499249"
> print(paste('standard deviation is', sd(mat)))
[1] "standard deviation is 0.282578449822536"
>
```

4. **Consider a problem where a user needs to group a set of people P who wants to apply for a loan at a bank. The main criterion for applying is the income of the person. Given I, which corresponds to the respective incomes of people in P. Group the people into a low-risk customer based on whether the customer's income is above 30000 (low risk) or not (High risk)**

```
P = scan(what='')
I = c(10000, 14000, 24000, 43000, 12323, 13414, 43212, 36000)

low_risk = character(0)
high_risk = character(0)

for(i in 1:length(I))
{
     if(I[i] < 30000){ high_risk = c(high_risk, P[i])}
     else { low_risk = c(low_risk, P[i]) }
}

print('High risk people')
high_risk
print('Low risk people')
low_risk
```

OUTPUT

```
> P = scan(what='')
1: Ayush
2: Aman
3: Arush
4: Anirban
5: Aparajita
6: Annie
7: Arpit
8: Arundhati
9:
Read 8 items
> low_risk = character(0)
> high_risk = character(0)
> for(i in 1:length(I))
+ {
+ if(I[i] < 30000){ high_risk = c(high_risk, P[i])}
+ else { low_risk = c(low_risk, P[i]) }
+ }
> print('High risk people')
[1] "High risk people"
> high_risk
[1] "Ayush"     "Aman"      "Arush"     "Aparajita" "Annie"
> print('Low risk people')
[1] "Low risk people"
> low_risk
[1] "Anirban"   "Arpit"     "Arundhati"
```

5. Assign the values TRUE, FALSE, FALSE, TRUE to a logical vector X and the values FALSE, TRUE, FALSE, TRUE to the logical vector Y. Perform element wise AND and OR, also find logical AND and logical OR of X and Y

```
# q5
X = c(TRUE, FALSE, FALSE, TRUE)
Y = c(FALSE, TRUE, FALSE, TRUE)

# (i)
print('element wise and of x and y')
X & Y
print('element wise or of x and y')
X | Y

# (ii)
a = X[0]
for(i in X) a = a && i

print(paste('Logical AND of X: ', a))

a = X[0]
for(i in X) a = a || i

print(paste('Logical OR of X: ', a))

a = Y[0]
for(i in Y) a = a && i

print(paste('Logical AND of Y: ', a))
```

```
a = Y[0]
for(i in Y) a = a || i

print(paste('Logical OR of Y: ', a))
```

OUTPUT

```
> print('element wise and of x and y')
[1] "element wise and of x and y"
> X & Y
[1] FALSE FALSE FALSE  TRUE
> print('element wise or of x and y')
[1] "element wise or of x and y"
> X | Y
[1]  TRUE  TRUE FALSE  TRUE
> a = X[0]
> for(i in X) a = a && i
>
> print(paste('Logical AND of X: ', a))
[1] "Logical AND of X:  FALSE"
> a = X[0]
> for(i in X) a = a || i
> print(paste('Logical OR of X: ', a))
[1] "Logical OR of X:  TRUE"
> a = Y[0]
> for(i in Y) a = a && i
>
> a = Y[0]
> for(i in Y) a = a || i
> print(paste('Logical OR of Y: ', a))
[1] "Logical OR of Y:  TRUE"
>
```

6. **Use the inbuilt iris and write r code for displaying the first few rows, displaying the structure of iris dataset, display the value that lies at the intersection of row 3 and column 4 and display the value that lies at the intersection of row 3 and columns 1 to 4**

```
# q6
data(iris)

# (a)
head(iris, 5)

# (b)
dim(iris)
summary(iris)
names(iris)

# (c)
x = iris
class(x)

x[3, 4]

# (d)
x[3, 1:4]
```

OUTPUT

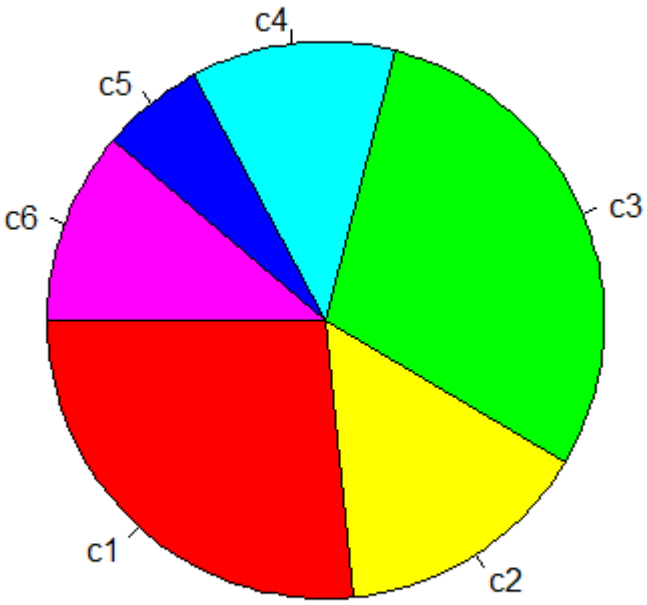
```
> head(iris, 5)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
> dim(iris)
[1] 150   5
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
 setosa    :50
 versicolor:50
 virginica :50


> names(iris)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
[5] "Species"
> x = iris
> class(x)
[1] "data.frame"
> x[3, 4]
[1] 0.2
> x[3, 1:4]
  Sepal.Length Sepal.Width Petal.Length Petal.Width
3          4.7         3.2          1.3         0.2
```

7. Given the name of chemical (c1, c2, …, c5) and amount produced, plot a pie chart with the initial angle 180 degree

 # q7

```
amt = c(90, 50, 100, 40, 20, 38)
chem = c('c1', 'c2', 'c3', 'c4', 'c5', 'c6')

pie(amt, label=chem, col=rainbow(length(amt)), init.angle=180)
```

OUTPUT



8. Write R code using a loop to print the required output

```
# q8


x = 1
while(x <= 7)
{
     if(x <= 3)
     {
          print('Four is greater than 3')
     }
     else if(x == 4)
     {
          print('Next')
     }
     else
     {
          print('Three is greater than two')
     }

     x = x + 1
}
```

OUTPUT

```
> amt = c(90, 50, 100, 40, 20, 38)
> chem = c('c1', 'c2', 'c3', 'c4', 'c5', 'c6')
> pie(amt, label=chem, col=rainbow(length(amt)), init.angle=180)
> x = 1
> while(x <= 7)
+ {
+ if(x <= 3)
+ {
+ print('Four is greater than 3')
+ }
+ else if(x == 4)
+ {
+ print('Next')
+ }
+ else
+ {
+ print('Three is greater than two')
+ }
+
+ x = x + 1
+ }
[1] "Four is greater than 3"
[1] "Four is greater than 3"
[1] "Four is greater than 3"
[1] "Next"
[1] "Three is greater than two"
[1] "Three is greater than two"
[1] "Three is greater than two"
```

9. Write a program in R to print the harmonic series and its sum

```
# q9

n = scan()
s = 0.0

for(i in 1:n)
{
     if(i < n)
     {
          cat('1/', i, ' + ')
          s = s + 1/i
     }
     if(i == n)
     {
          cat('1/', i)
          s = s + 1/i
     }
}

cat('sum ', s)
```

OUTPUT

```
> n = scan()
1: 7
2:
Read 1 item
> s = 0.0
>
> for(i in 1:n)
+ {
+ if(i < n)
+ {
+ cat('1/', i, ' + ')
+ s = s + 1/i
+ }
+ if(i == n)
+ {
+ cat('1/', i)
+ s = s + 1/i
+ }
+ }
1/ 1  + 1/ 2  + 1/ 3  + 1/ 4  + 1/ 5  + 1/ 6  + 1/ 7>
> cat('sum ', s)
sum  2.592857>
```

10. Write a nested loop where the outer for() loop increments ‘a’ 3 times and ‘b’ 4 times. The break statement exits the loop after 2 increments. The nested loop prints the value of elements a and b

```
# q10


a = 0
b = 0


for(i in 1:3)
{
     a = a + 1
     for(j in 1:4)
     {
          b = b + 1
          if(j == 2)
          {
               break
          }
          print(paste('a = ',a))
          print(paste('b = ',b))
     }
}
```

OUTPUT

```
> b = 0
>
> for(i in 1:3)
+ {
+ a = a + 1
+ for(j in 1:4)
+ {
+ b = b + 1
+ if(j == 2)
+ {
+ break
+ }
+ print(paste('a = ',a))
+ print(paste('b = ',b))
+ }
+ }
[1] "a =  1"
[1] "b =  1"
[1] "a =  2"
[1] "b =  3"
[1] "a =  3"
[1] "b =  5"
```

11. Write a R program to create a student data frame, replace a value across the entire data frame and replace multiple values across the dataframe

```
# q11


sid = c(1,2, 3, 4, 5, 6)
age = c(22, 23, 22, 21, 21, 24)
pointer = c(8, 9, 8, 7, 6, 8)
dept = c('mca', 'mca', 'mca', 'mca', 'mca', 'mca')

student_df = data.frame(sid, age, pointer, dept)


student_df[2, 3] = 6.6


student_df[student_df == 'mca'] = 'cse'
student_df
```

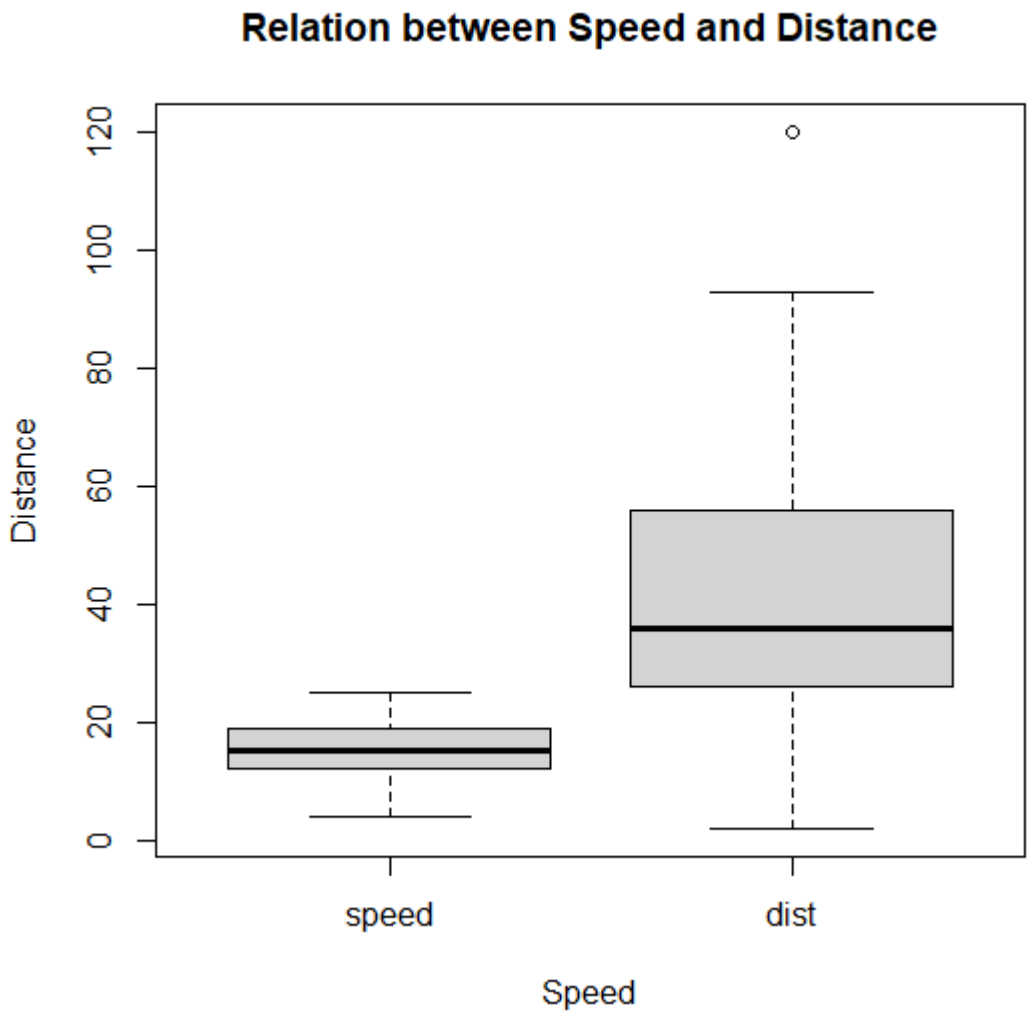
OUTPUT

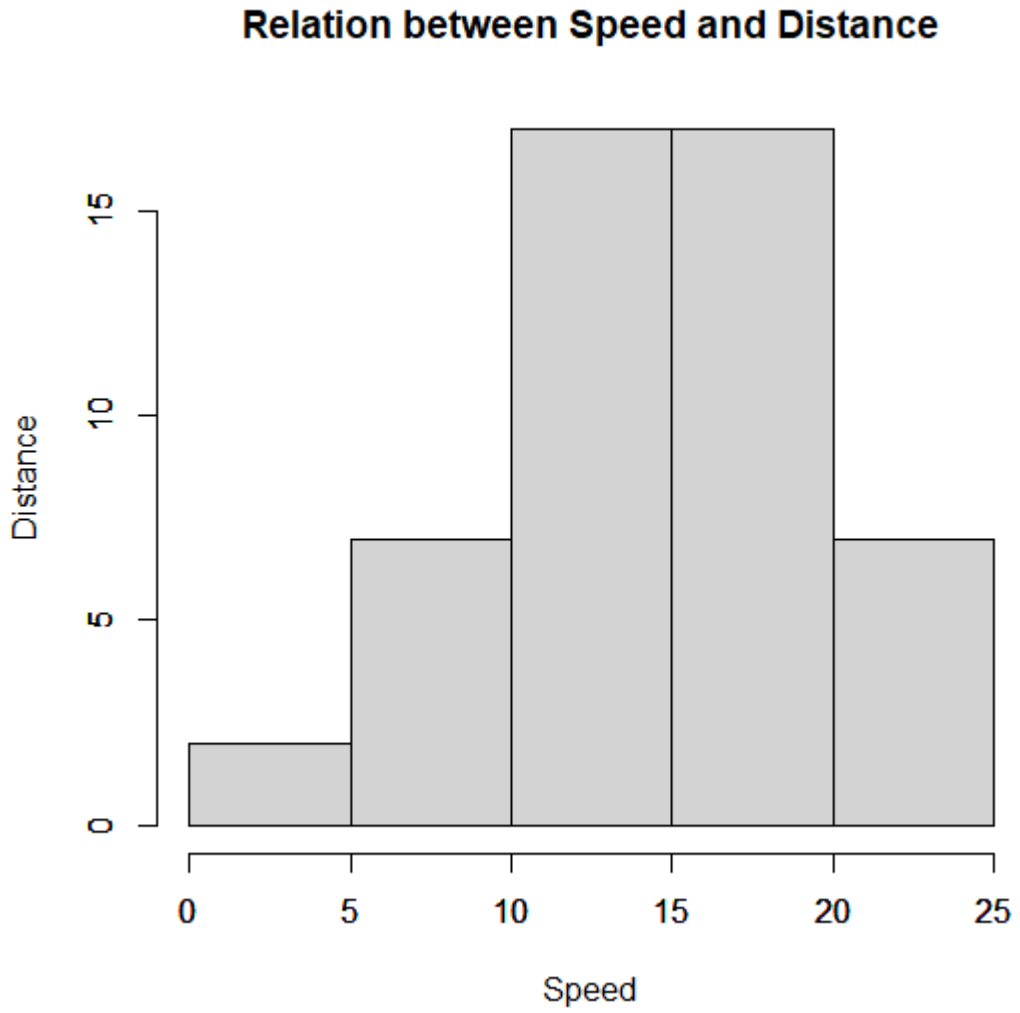
```
> sid = c(1,2, 3, 4, 5, 6)
> age = c(22, 23, 22, 21, 21, 24)
> pointer = c(8, 9, 8, 7, 6, 8)
> dept = c('mca', 'mca', 'mca', 'mca', 'mca', 'mca')
> student_df = data.frame(sid, age, pointer, dept)
> student_df[2, 3] = 6.6
> student_df[student_df == 'mca'] = 'cse'
> student_df
  sid age pointer dept
1   1  22     8.0  cse
2   2  23     6.6  cse
3   3  22     8.0  cse
4   4  21     7.0  cse
5   5  21     6.0  cse
6   6  24     8.0  cse
```

12. Use inbuilt cars data frame to create a box plot as well as histogram

```
# q12


data(cars)

head(cars, 10)

a = cars

boxplot(a, main='Relation between Speed and Distance',
xlab='Speed', ylab='Distance')

hist(a$speed, main='Relation between Speed and Distance',
xlab='Speed', ylab='Distance')
```

OUTPUT




Relation between Speed and Distance
Distance
0
5
10
15
0
5
10
15
20
25
Speed