22021BCS0175
Y.Hazarathayya

LAB - 2

DATA WAREHOUSING & DATA MINING

1)Code:

```
df1 <- data.frame(student_name = c('Alice', 'Bob', 'Charlie'),
                  marks = c(85, 92, 78))
df2 <- data.frame(student_name = c('David', 'Eva', 'Frank'),
                  marks = c(88, 95, 80))
result_df <- merge(df1, df2, by = 'student_name', all = TRUE)
print(result_df)
```

Result:

```
> # Creating the first data frame
> df1 <- data.frame(student_name = c('Alice', 'Bob', 'Charlie'),
+                   marks = c(85, 92, 78))
> # Creating the second data frame
> df2 <- data.frame(student_name = c('David', 'Eva', 'Frank'),
+                   marks = c(88, 95, 80))
> # Merging the data frames on 'student_name'
> result_df <- merge(df1, df2, by = 'student_name', all = TRUE)
> # Displaying the content of the resultant data frame
> print(result_df)
  student_name marks.x marks.y
1        Alice      85      NA
2          Bob      92      NA
3      Charlie      78      NA
4        David      NA      88
5          Eva      NA      95
6        Frank      NA      80
>
```

2)Code:

```
m <- c(1, 2, 3, 4, 5)
n <- c(6, 7, 8, 9, 10)
p <- c(11, 12, 13, 14, 15)

X <- matrix(c(m, n, p), nrow = 5, byrow = FALSE)
print(X)
```

Result:

```
> # Create three integer vectors with 5 elements each
> m <- c(1, 2, 3, 4, 5)
> n <- c(6, 7, 8, 9, 10)
> p <- c(11, 12, 13, 14, 15)
>
> # Combine the vectors to create a 5x5 matrix
> X <- matrix(c(m, n, p), nrow = 5, byrow = FALSE)
>
> # Print the resulting matrix
> print(X)
     [,1] [,2] [,3]
[1,]    1    6   11
[2,]    2    7   12
[3,]    3    8   13
[4,]    4    9   14
[5,]    5   10   15
```

3)

```
A <- c(10, 4, -2, 16, 17, 15, 12, 14, 15)
Y <- A[A > 10]
print("Vector A:")
print(A)
print("Vector Y (elements of A > 10):")
print(Y)
```

Result:

```
> # Create a vector A
> A <- c(10, 4, -2, 16, 17, 15, 12, 14, 15)
>
> # Create a vector Y containing elements of A greater than 10
> Y <- A[A > 10]
>
> # Print the vectors A and Y
> print("Vector A:")
[1] "Vector A:"
> print(A)
[1] 10  4 -2 16 17 15 12 14 15
>
> print("Vector Y (elements of A > 10):")
[1] "Vector Y (elements of A > 10):"
> print(Y)
[1] 16 17 15 12 14 15
```

4)Code

```
sum_divisible_by_5_and_7 <- 0
for (i in 1:100) {
  if (i %% 5 == 0 && i %% 7 == 0) {
    sum_divisible_by_5_and_7 <- sum_divisible_by_5_and_7
+ i
  }
}
print(paste("Sum of numbers from 1 to 100 divisible by both 5
and 7:", sum_divisible_by_5_and_7))
```

Result:

```
> sum_divisible_by_5_and_7 <- 0
>
> # Loop through numbers from 1 to 100
> for (i in 1:100) {
+   # Check if the number is divisible by both 5 and 7
+   if (i %% 5 == 0 && i %% 7 == 0) {
+     # If true, add it to the sum
+     sum_divisible_by_5_and_7 <- sum_divisible_by_5_and_7 + i
+   }
+ }
>
> # Print the sum
> print(paste("Sum of numbers from 1 to 100 divisible by both 5 and 7:",
sum_divisible_by_5_and_7))
[1] "Sum of numbers from 1 to 100 divisible by both 5 and 7: 105"
```

5)Code

```
doubleOdd <- function(input_vector) {
  result_vector <- sapply(input_vector, function(x) ifelse(x %%
2 != 0, x * 2, x))
  return(result_vector)
}

input_vector <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
result_vector <- doubleOdd(input_vector)
print("Original Vector:")
print(input_vector)
print("Modified Vector (doubling odd numbers):")
print(result_vector)
```

```
6)determineQuadrant <- function(angle) {
  if (0 <= angle && angle < 90) {
    cat(angle, "degrees is in Quadrant 1.\n")
  } else if (90 <= angle && angle < 180) {
    cat(angle, "degrees is in Quadrant 2.\n")
  } else if (180 <= angle && angle < 270) {
    cat(angle, "degrees is in Quadrant 3.\n")
  } else if (270 <= angle && angle < 360) {
    cat(angle, "degrees is in Quadrant 4.\n")
  } else {
    cat("Invalid angle. Please enter a value between 0 and
359.\n")
  }
}

# Example usage
angle <- 45
determineQuadrant(angle)

angle <- 120
determineQuadrant(angle)

angle <- 200
determineQuadrant(angle)

angle <- 300
determineQuadrant(angle)
```

```
> # Example usage
> angle <- 45
> determineQuadrant(angle)
45 degrees is in Quadrant 1.
>
> angle <- 120
> determineQuadrant(angle)
120 degrees is in Quadrant 2.
>
> angle <- 200
> determineQuadrant(angle)
200 degrees is in Quadrant 3.
>
> angle <- 300
> determineQuadrant(angle)
300 degrees is in Quadrant 4.
```

7)

```
calculateDoubleSum <- function(n) {
  result <- 0

  for (i in 1:n) {
    for (r in 1:i) {
      result <- result + (r^2) / (10 + 4 * r^3)
    }
  }

  return(result)
}

# Example usage
n_value <- 3  # You can change this to any positive integer
result <- calculateDoubleSum(n_value)
cat("The double sum for n =", n_value, "is:", result, "\n")
```

```
> calculateDoubleSum <- function(n) {
+   result <- 0
+
+   for (i in 1:n) {
+     for (r in 1:i) {
+       result <- result + (r^2) / (10 + 4 * r^3)
+     }
+   }
+
+   return(result)
+ }
>
> # Example usage
> n_value <- 3  # You can change this to any positive integer
> result <- calculateDoubleSum(n_value)
> cat("The double sum for n =", n_value, "is:", result, "\n")
The double sum for n = 3 is: 0.4810331
>
```