Lab1

Atanu Giri

2025-01-21

Problem 1: Vector Manipulations

1. Create a numeric vector v of length 10 containing the first 10 positive integers.
2. Add these three numbers to vector v: 18, 20 and -15.
3. Remove the 3rd and 5th element of v.
4. Extract the 4th to 7th elements of v.
5. Replace the even numbers in v with NA.
6. Calculate the sum of all non-NA elements in v.

```
(v = 1:10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
(v = c(v, 18, 20, -15))
```

```
##  [1]   1   2   3   4   5   6   7   8   9  10  18  20 -15
```

```
(v = v[-c(3,5)])
```

```
##  [1]   1   2   4   6   7   8   9  10  18  20 -15
```

```
(v[4:7])
```

```
## [1] 6 7 8 9
```

```
v[v %% 2 == 0] = NA
v
```

```
##  [1]   1  NA  NA  NA   7  NA   9  NA  NA  NA -15
```

```
(sum(v,na.rm=TRUE))
```

```
## [1] 2
```

Problem 2: Matrix Manipulations

1. Create a 3×3 matrix M with elements from 1 to 9 filled row-wise.
2. Extract the second row of the matrix.
3. Calculate the sum of each column in the matrix.
4. Add a new row [10, 11, 12] to the matrix.
5. Multiply the resulting matrix by 2.
6. Multiply the resulting matrix by a 3×4 matrix, where elements are randomly generated from the standard normal distribution, N(0,1). Call the new matrix P.
7. Find the determinant of P

```
(M = matrix(1:9, nrow = 3, byrow = TRUE))
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

```
(M[2,])
```

```
## [1] 4 5 6
```

```
(apply(M, 2, sum))
```

```
## [1] 12 15 18
```

```
(M = rbind(M, c(10,11,12)))
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
## [4,]   10   11   12
```

```
(M = M*2)
```

```
##      [,1] [,2] [,3]
## [1,]    2    4    6
## [2,]    8   10   12
## [3,]   14   16   18
## [4,]   20   22   24
```

```
M2 = matrix(rnorm(12), nrow = 3)
(P = M %*% M2)
```

```
##           [,1]       [,2]      [,3]      [,4]
## [1,] -12.09236 0.02945815 -10.51835  9.502448
## [2,] -25.38483 0.35079719 -24.06244 16.252731
## [3,] -38.67729 0.67213623 -37.60652 23.003013
## [4,] -51.96975 0.99347527 -51.15061 29.753295
```

```r
det(P)
```

```
## [1] -3.829148e-29
```

Problem 3: Conditional Statements

1. Write an R script that checks whether a given number x is positive, negative, or zero and prints an appropriate message. Example: If x = -3, the output should be: "x = -3 is negative."
2. Create a function check_even_odd that takes a number as input and returns whether it is "Even" or "Odd".

```r
x = -3

if (x > 0) {
  cat("x =", x, "is positive.\n")
} else if (x < 0) {
  cat("x =", x, "is negative.\n")
} else {
  cat("x =", x, "is zero.\n")
}
```

```
## x = -3 is negative.
```

```r
check_even_odd = function(x)
  if (x %% 2 == 0){
    return("Even")
  } else {
    return("Odd")
  }

result1 <- check_even_odd(4)
print(result1)
```

```
## [1] "Even"
```

Problem 4: Loops

For Loop

1. Write a for loop to print the squares of numbers from 1 to 10.
2. Use a for loop to calculate the factorial of 5.

```r
for (i in 1:10){
  print(i^2)
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
## [1] 64
## [1] 81
## [1] 100
```

```r
s = 1
for (i in 1:5){
  s = s*i
}
print(s)
```

```
## [1] 120
```

While Loop

1. Write a while loop to print numbers from 1 to 10.
2. Write a while loop to calculate the sum of numbers from 1 to 50.

```r
i = 1
while (i < 11) {
  print(i)
  i = i+1
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

```r
s = 0
i = 1
while (i < 51) {
  s = s+i
  i = i+1
}
print(s)
```

```
## [1] 1275
```

Problem 5: User-Defined Functions

1. Write a function sum_and_product that takes two arguments and returns their sum and product as a list. Example: For inputs 2 and 3, the output should be: list(sum = 5, product = 6).
2. Create a function generate_fibonacci that takes a single argument n and returns the first n Fibonacci numbers. Example: For n = 5, the output should be: 1, 1, 2, 3, 5.

```
sum_and_product = function(x,y){
  sum = x+y
  product = x*y
  return(list(sum = sum, product = product))
}

output = sum_and_product(2,3)
print(output)
```

```
## $sum
## [1] 5
##
## $product
## [1] 6
```

```
generate_fibonacci <- function(n) {
  if (n <= 0) {
    stop("n must be a positive integer.")
  }

  # Initialize a vector to store Fibonacci numbers
  fibonacci <- numeric(n)

  if (n >= 1) fibonacci[1] <- 1
  if (n >= 2) fibonacci[2] <- 1

  for (i in 3:n) {
    fibonacci[i] <- fibonacci[i - 1] + fibonacci[i - 2]
  }

  return(fibonacci)
}

output = generate_fibonacci(5)
print(output)
```

```
## [1] 1 1 2 3 5
```