

PRAKTIKUM PEMODELAN STATISTIKA

MODUL 1



Disusun oleh :

Nama : Fidelia Ping

NIM : 245410012

Kelas : Informatika 1

**PROGRAM STUDI INFORMATIKA
PROGRAM SARJANA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA
YOGYAKARTA
2025**

MODUL 1

PEMROGRAMAN R MENGGUNAKAN GOOGLE COLAB

A. TUJUAN PRAKTIKUM

Memahami cara menjalankan R di Google Colab. Menguasai dasar-dasar pemrograman R seperti variabel, tipe data, dan struktur data

B. PEMBAHASAN LISTING

Praktikum

- Membuat dan Memanipulasi Vektor.

Membuat Vektor dengan fungsi c(), seq(), atau rep().

```
▶ # Membuat vektor numerik
nilai <- c(80, 85, 90, 95, 100)
cat("Vektor nilai:", nilai, "\n")

# Membuat vektor karakter
nama <- c("Ali", "Budi", "Cici")
cat("Vektor nama:", nama, "\n")

# Membuat vektor logis
lulus <- c(TRUE, TRUE, FALSE)
cat("Vektor lulus:", lulus, "\n")

# Menggunakan seq() untuk urutan
urutan <- seq(from = 1, to = 10, by = 2)
cat("Vektor urutan:", urutan, "\n")

# Menggunakan rep() untuk pengulangan
ulang <- rep(5, times = 3)
cat("Vektor pengulangan:", ulang, "\n")
```

output:

→ Vektor nilai: 80 85 90 95 100
Vektor nama: Ali Budi Cici
Vektor lulus: TRUE TRUE FALSE
Vektor urutan: 1 3 5 7 9
Vektor pengulangan: 5 5 5

Pembahasan : Program tersebut menunjukkan cara membuat berbagai jenis vektor di R, yaitu vektor numerik, karakter, dan logis menggunakan fungsi c(), membuat urutan angka dengan seq(), serta pengulangan nilai dengan rep(), di mana hasilnya menampilkan kumpulan data berbeda seperti angka, teks, nilai logika, urutan bilangan, dan pengulangan nilai dalam satu bentuk struktur vektor.

Melakukan operasi aritmatika langsung pada vektor

```
▶ # Operasi aritmatika
nilai <- c(80, 85, 90, 95, 100)
nilai_tambah <- nilai + 5 # Menambah 5 ke setiap elemen
cat("Nilai setelah ditambah 5:", nilai_tambah, "\n")

# Operasi antar vektor
bonus <- c(1, 2, 3, 4, 5)
total <- nilai + bonus
cat("Total nilai dengan bonus:", total, "\n")

# Operasi logis
tinggi <- nilai > 85
cat("Nilai > 85:", tinggi, "\n")
```

output:

→ Nilai setelah ditambah 5: 85 90 95 100 105
Total nilai dengan bonus: 81 87 93 99 105
Nilai > 85: FALSE FALSE TRUE TRUE TRUE

Pembahasan : Program ini menunjukkan operasi dasar pada vektor di R, yaitu penambahan angka pada seluruh elemen, penjumlahan antar vektor dengan elemen sepadan, dan perbandingan logis untuk menghasilkan vektor bernilai TRUE atau FALSE.

Pengindeksan dan Filtering vektor (pada R dimulai dari 1)

```
▶ # Pengindeksan
  nilai <- c(80, 85, 90, 95, 100)
  cat("Elemen kedua:", nilai[2], "\n")
  cat("Elemen 1 sampai 3:", nilai[1:3], "\n")

  # Filtering
  nilai_tinggi <- nilai=nilai > 85
  cat("Nilai > 85:", nilai_tinggi, "\n")
```

➡ Elemen kedua: 85
Elemen 1 sampai 3: 80 85 90
Nilai > 85: 90 95 100

Output:

Pembahasan : Program ini menunjukkan cara mengakses elemen tertentu dari vektor menggunakan indeks dan cara melakukan penyaringan (filtering) untuk menampilkan hanya nilai yang memenuhi kondisi tertentu, seperti nilai lebih besar dari 85.

- Membuat dan Memanipulasi Factor

Membuat faktor tidak terurut dengan factor()

```
▶ # Membuat vektor karakter
  jenis_kelamin <- c("Laki-laki", "Perempuan", "Laki-laki", "Perempuan", "Laki-laki")

  # Membuat faktor tidak terurut
  jk_faktor <- factor(jenis_kelamin)
  cat("Faktor jenis kelamin:", jk_faktor, "\n")
  cat("Level faktor:", levels(jk_faktor), "\n")
  cat("Jumlah level:", nlevels(jk_faktor), "\n")

  # Mengubah urutan level
  jk_faktor_urut <- factor(jenis_kelamin, levels = c("Perempuan", "Laki-laki"))
  cat("Level dengan urutan khusus:", levels(jk_faktor_urut), "\n")
```

➡ Faktor jenis kelamin: 1 2 1 2 1
 Level faktor: Laki-laki Perempuan
 Jumlah level: 2

Output: Level dengan urutan khusus: Perempuan Laki-laki

Pembahasan : Program ini memperlihatkan cara membuat vektor karakter, mengubahnya menjadi faktor untuk mengelompokkan data kategorikal, menampilkan level dan jumlah kategorinya, serta mengatur urutan level sesuai keinginan pengguna.

Membuat Faktor terurut menggunakan ordered() atau factor(ordered = TRUE)

```
▶ # Membuat vektor untuk tingkat pendidikan
  pendidikan <- c("SD", "SMP", "SMA", "SMP", "Sarjana", "SMA")

  # Membuat faktor terurut
  pend_faktor <- ordered(pendidikan, levels = c("SD", "SMP", "SMA", "Sarjana"))
  cat("Faktor pendidikan terurut:", pend_faktor, "\n")
  cat("Level faktor:", levels(pend_faktor), "\n")
  cat("Apakah terurut?", is.ordered(pend_faktor), "\n")
```

➡ Faktor pendidikan terurut: 1 2 3 2 4 3
 Level faktor: SD SMP SMA Sarjana

Output: Apakah terurut? TRUE

Pembahasan : Program ini menunjukkan cara membuat faktor terurut di R untuk data kategorikal seperti tingkat pendidikan, menampilkan urutan levelnya, dan memverifikasi bahwa faktor tersebut memiliki tingkatan logis dari SD hingga Sarjana.

Melakukan operasi pada Faktor

- Konversi ke karakter: as.character().
- Mengubah level: relevel().
- Menggabungkan level: droplevels() atau factor() baru.

```

▶ # Membuat vektor karakter
jenis_kelamin <- c("Laki-laki", "Perempuan", "Laki-laki", "Perempuan", "Laki-laki")

# Membuat faktor tidak terurut
jk_faktor <- factor(jenis_kelamin)

# Konversi faktor ke karakter
cat("Faktor ke karakter:", as.character(jk_faktor), "\n")

# Mengubah level referensi
jk_relevel <- relevel(jk_faktor, ref = "Perempuan")
cat("Level setelah relevel:", levels(jk_relevel), "\n")

# Menghapus level tidak terpakai
faktor_lengkap <- factor(c("A", "B", "C", "A"))
faktor_diseder <- droplevels(faktor_lengkap)
cat("Level setelah drop:", levels(faktor_diseder), "\n")

```

→ Faktor ke karakter: Laki-laki Perempuan Laki-laki Perempuan Laki-laki
 Level setelah relevel: Perempuan Laki-laki
Output : Level setelah drop: A B C

Pembahasan : Program ini menjelaskan cara mengubah faktor menjadi karakter, mengatur level referensi dengan relevel(), serta menyederhanakan faktor dengan menghapus level yang tidak digunakan menggunakan droplevels().

- Membuat dan Memanipulasi Frame

Membuat Data frame dibuat dengan data.frame(), dengan kolom sebagai vektor.

```

▶ # Membuat data frame
data_siswa <- data.frame(
  Nama = c("Ali", "Budi", "Cici", "Dewi"),
  Nilai = c(80, 85, 90, 75),
  Kelas = factor(c("A", "B", "A", "B")),
  Usia = c(20, 21, 19, 20)
)
cat("Data frame siswa:\n")
print(data_siswa)

# Melihat struktur data frame
str(data_siswa)

```

→ Data frame siswa:
 Nama Nilai Kelas Usia
 1 Ali 80 A 20
 2 Budi 85 B 21
 3 Cici 90 A 19
 4 Dewi 75 B 20
 'data.frame': 4 obs. of 4 variables:
 \$ Nama : chr "Ali" "Budi" "Cici" "Dewi"
 \$ Nilai: num 80 85 90 75
 \$ Kelas: Factor w/ 2 levels "A","B": 1 2 1 2
 \$ Usia : num 20 21 19 20

Output :
Pembahasan : Program ini membuat data frame berisi informasi siswa dengan berbagai tipe data (karakter, numerik, dan faktor), lalu menggunakan str() untuk menampilkan struktur dan tipe data dari setiap kolom dalam tabel tersebut.

Mengakses dan memodifikasi data frame

```
▶ # Mengakses kolom
cat("Kolom Nilai:", data_siswa$Nilai, "\n")

# Mengakses baris
cat("Baris pertama:\n")
print(data_siswa[1, ])

# Mengakses elemen tertentu
cat("Nilai Budi:", data_siswa[2, "Nilai"], "\n")

# Menambahkan kolom baru
data_siswa$Lulus <- data_siswa$Nilai >= 80
cat("Data frame dengan kolom Lulus:\n")
print(data_siswa)

# Menggunakan subset() untuk filtering
data_lulus <- subset(data_siswa, Lulus == TRUE)
cat("Siswa yang lulus:\n")
print(data_lulus)
```

Output :

```
→ Kolom Nilai: 80 85 90 75
Baris pertama:
  Nama Nilai Kelas Usia
1  Ali    80     A   20
Nilai Budi: 85
Data frame dengan kolom Lulus:
  Nama Nilai Kelas Usia Lulus
1  Ali    80     A   20  TRUE
2  Budi   85     B   21  TRUE
3  Cici   90     A   19  TRUE
4  Dewi   75     B   20 FALSE
Siswa yang lulus:
  Nama Nilai Kelas Usia Lulus
1  Ali    80     A   20  TRUE
2  Budi   85     B   21  TRUE
3  Cici   90     A   19  TRUE
```

Pembahasan : Program ini menunjukkan cara mengakses kolom, baris, dan elemen tertentu dalam data frame, menambahkan kolom logis baru berdasarkan kondisi, serta memfilter data menggunakan fungsi subset() untuk menampilkan siswa yang memenuhi syarat kelulusan.

Manipulasi Data Frame dengan dplyr

```
▶ # Install dan load dplyr
install.packages("dplyr")
library(dplyr)

# Filtering
nilai_tinggi <- data_siswa %>% filter(Nilai >= 85)
cat("Siswa dengan nilai >= 85:\n")
print(nilai_tinggi)

# Pengelompokan dan agregasi
rata_kelas <- data_siswa %>% group_by(Kelas) %>% summarise(Rata_rata = mean(Nilai))
cat("Rata-rata nilai per kelas:\n")
print(rata_kelas)
```

```

→ Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Siswa dengan nilai >= 85:
  Nama Nilai Kelas Usia Lulus
  1 Budi    85     B   21 TRUE
  2 Cici    90     A   19 TRUE
Rata-rata nilai per kelas:
# A tibble: 2 × 2
  Kelas Rata_rata
  <fct>     <dbl>
  1 A          85
  2 B          80

```

Output :

Pembahasan : Program ini menggunakan paket dplyr untuk memfilter siswa dengan nilai tinggi dan menghitung rata-rata nilai berdasarkan kelas melalui operasi filter(), group_by(), dan summarise() yang efisien dan mudah dibaca.

Menggabungkan Data Frame

```

▶ # Membuat data frame tambahan
  data_tambahan <- data.frame(
    Nama = c("Ali", "Budi", "Eka"),
    Peminatan = c("Matematika", "Fisika", "Kimia"))

  # Menggabungkan dengan merge
  data_gabung <- merge(data_siswa, data_tambahan, by = "Nama", all.x = TRUE)
  cat("Data frame gabungan:\n")
  print(data_gabung)

```

→ Data frame gabungan:

	Nama	Nilai	Kelas	Usia	Lulus	Peminatan
1	Ali	80	A	20	TRUE	Matematika
2	Budi	85	B	21	TRUE	Fisika
3	Cici	90	A	19	TRUE	<NA>
4	Dewi	75	B	20	FALSE	<NA>

Output :

Pembahasan : Program ini menggabungkan dua data frame menggunakan merge() berdasarkan kolom Nama dengan opsi all.x = TRUE, sehingga menghasilkan gabungan bertipe *left join* yang menambahkan kolom Peminatan ke data siswa sambil mempertahankan semua baris dari tabel utama.

- Membuat dan Memanipulasi Matriks dan Array

Membuat Matriks

```
▶ # Membuat matriks 2x3
matriks <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3, byrow = FALSE)
cat("Matriks 2x3 (isi per kolom):\n")
print(matriks)

# Membuat matriks dengan pengisian per baris
matriks_byrow <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3, byrow = TRUE)
cat("Matriks 2x3 (isi per baris):\n")
print(matriks_byrow)

# Menambahkan nama baris dan kolom
rownames(matriks) <- c("Baris1", "Baris2")
colnames(matriks) <- c("Kolom1", "Kolom2", "Kolom3")
cat("Matriks dengan nama:\n")
print(matriks)
→ Matriks 2x3 (isi per kolom):
 [,1] [,2] [,3]
 [1,]    1    3    5
 [2,]    2    4    6
 Matriks 2x3 (isi per baris):
 [,1] [,2] [,3]
 [1,]    1    2    3
 [2,]    4    5    6
 Matriks dengan nama:
 Kolom1 Kolom2 Kolom3
 Baris1    1    3    5
 Baris2    2    4    6
```

Output :

Pembahasan : Program ini menunjukkan cara membuat matriks di R dengan pengisian per kolom dan per baris menggunakan fungsi matrix(), serta menambahkan nama baris dan kolom untuk memperjelas struktur data.

Operasi Matriks

```
▶ # Transpos matriks
cat("Transpos matriks:\n")
print(t(matriks))

# Perkalian matriks
matriks2 <- matrix(c(1, 0, 0, 1, 0, 0), nrow = 3, ncol = 2)
hasil_kali <- matriks %*% matriks2
cat("Hasil perkalian matriks:\n")
print(hasil_kali)

# Invers matriks (khusus matriks persegi)
matriks_persegi <- matrix(c(4, 3, 3, 2), nrow = 2, ncol = 2)
invers <- solve(matriks_persegi)
cat("Invers matriks persegi:\n")
print(invers)
```

Output :

```

→ Transpos matriks:
      Baris1 Baris2
Kolom1      1      2
Kolom2      3      4
Kolom3      5      6
Hasil perkalian matriks:
      [,1] [,2]
Baris1      1      1
Baris2      2      2
Invers matriks persegi:
      [,1] [,2]
[1,]    -2     3
[2,]     3    -4

```

Pembahasan : Program ini menunjukkan cara melakukan operasi dasar pada matriks di R yaitu menghitung transpos, melakukan perkalian antar matriks, dan mencari invers dari matriks persegi menggunakan fungsi t(), %*%, dan solve().

Membuat Array

```

▶ # Membuat array 3D (2x3x2)
array_data <- array(1:12, dim = c(2, 3, 2))
cat("Array 3D:\n")
print(array_data)

# Menambahkan nama dimensi
dimnames(array_data) <- list(
  Baris = c("B1", "B2"),
  Kolom = c("K1", "K2", "K3"),
  Dimensi = c("D1", "D2")
)
cat("Array dengan nama dimensi:\n")
print(array_data)

```

Output :

```

→ Array 3D:
, , 1
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
, , 2
      [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12
Array dengan nama dimensi:
, , Dimensi = D1
      Kolom
Baris K1 K2 K3
  B1  1  3  5
  B2  2  4  6
, , Dimensi = D2
      Kolom
Baris K1 K2 K3
  B1  7  9 11
  B2  8 10 12

```

Pembahasan : Program ini membuat array 3 dimensi berukuran $2 \times 3 \times 2$ berisi angka 1–12 dan memberi nama pada setiap dimensi (baris, kolom, dan lapisan) untuk mempermudah identifikasi data dalam struktur multidimensi.

Mengakses dan memodifikasi matriks/array

```
▶ # Mengakses elemen matriks
  cat("Elemen [1,2] matriks:", matriks[1, 2], "\n")

  # Mengakses baris/kolom
  cat("Baris pertama:\n")
  print(matriks[1, ])
  cat("Kolom kedua:\n")
  print(matriks[, 2])

  # Mengakses elemen array
  cat("Elemen [1,2,1] array:", array_data[1, 2, 1], "\n")

  # Memodifikasi elemen
  matriks[1, 1] <- 10
  cat("Matriks setelah modifikasi:\n")
  print(matriks)
```

Elemen [1,2] matriks: 3
Baris pertama:
Kolom1 Kolom2 Kolom3
1 3 5
Kolom kedua:
Baris1 Baris2
3 4
Elemen [1,2,1] array: 3
Matriks setelah modifikasi:
Kolom1 Kolom2 Kolom3
Baris1 10 3 5
Baris2 2 4 6

Output :

Pembahasan : Program ini menunjukkan cara mengakses baris, kolom, dan elemen tertentu dari matriks serta array 3D, sekaligus memodifikasi nilai elemen menggunakan indeks baris, kolom, dan dimensi pada struktur data di R.

- Membuat dan Memanipulasi List

Membuat List dengan list(), dengan elemen bernama atau tanpa nama.

```
▶ # Membuat list dengan elemen heterogen
  data_list <- list(
    Nama = c("Ali", "Budi", "Cici"),
    Nilai = c(80, 85, 90),
    Kelas = factor(c("A", "B", "A")),
    Matriks = matrix(1:4, nrow = 2, ncol = 2)
  )
  cat("List data:\n")
  print(data_list)

  # Struktur list
  str(data_list)
```

List data:
\$Nama
[1] "Ali" "Budi" "Cici"
\$Nilai
[1] 80 85 90
\$Kelas
[1] A B A
Levels: A B
\$Matriks
[,1] [,2]
[1,] 1 3
[2,] 2 4

List of 4
\$ Nama : chr [1:3] "Ali" "Budi" "cici"
\$ Nilai : num [1:3] 80 85 90
\$ Kelas : Factor w/ 2 levels "A","B": 1 2 1
\$ Matriks: int [1:2, 1:2] 1 2 3 4

output :

Pembahasan : Program ini membuat list berisi berbagai jenis data seperti vektor, faktor, dan matriks lalu menampilkan strukturnya menggunakan str() untuk memperlihatkan tipe dan isi setiap elemen dalam list.

Mengakses Elemen List dengan [[]], \$, atau []

```
▶ # Mengakses dengan nama
  cat("Nama:", data_list$Nama, "\n")

  # Mengakses dengan indeks
  cat("Nilai (indeks 2):", data_list[[2]], "\n")

  # Mengakses sublist
  sublist <- data_list[1:2]
  cat("Sublist (Nama dan Nilai):\n")
  print(sublist)

  # Mengakses elemen dalam matriks
  cat("Elemen matriks [1,2]:", data_list$Matriks[1, 2], "\n") output :
```

→ Nama: Ali Budi Cici
Nilai (indeks 2): 80 85 90
Sublist (Nama dan Nilai):
\$Nama
[1] "Ali" "Budi" "Cici"

\$Nilai
[1] 80 85 90

Elemen matriks [1,2]: 3

Pembahasan : Program ini menunjukkan cara mengakses data dalam list menggunakan nama elemen (\$), indeks ([[]]), membuat sublist ([]), serta mengambil nilai dari elemen bersarang seperti matriks di dalam list.

Memodifikasi List, Tambah, ubah, atau hapus elemen list.

```
▶ # Menambahkan elemen baru
  data_list$Usia <- c(20, 21, 19)
  cat("List setelah menambah Usia:\n")
  print(data_list)

  # Mengubah elemen
  data_list$Nama[2] <- "Bambang"
  cat("List setelah mengubah Nama[2]:\n")
  print(data_list)

  # Menghapus elemen
  data_list$Matriks <- NULL
  cat("List setelah menghapus Matriks:\n")
  print(data_list)
```

output :

```
List setelah menambah Usia:
```

```
$Nama
[1] "Ali" "Budi" "Cici"
```

```
$Nilai
[1] 80 85 90
```

```
$Kelas
[1] A B A
Levels: A B
```

```
$Matriks
[,1] [,2]
[1,]    1    3
[2,]    2    4
```

```
$Usia
[1] 20 21 19
```

```
List setelah mengubah Nama[2]:
$Nama
[1] "Ali"      "Bambang" "Cici"
```

```
$Nilai
[1] 80 85 90
```

```
$Kelas
[1] A B A
Levels: A B
```

```
$Usia
[1] 20 21 19
```

```
List setelah menghapus Matriks:
$Nama
[1] "Ali"      "Bambang" "Cici"
```

```
$Nilai
[1] 80 85 90
```

```
$Kelas
[1] A B A
Levels: A B
```

```
$Usia
[1] 20 21 19
```

Pembahasan : Program ini memperlihatkan cara memanipulasi list di R dengan menambahkan elemen baru, mengubah isi elemen tertentu, dan menghapus elemen yang tidak diperlukan menggunakan operator \$ dan penugasan NULL.

Menggabungkan List menggunakan c() atau list() untuk menggabungkan list.

```
▶ # Membuat list kedua
list_tambahan <- list(
  Jurusan = c("Matematika", "Fisika", "Kimia"),
  Status = c(TRUE, FALSE, TRUE)
)

# Menggabungkan list
list_gabung <- c(data_list, list_tambahan)
cat("List gabungan:\n")
print(list_gabung)
```

output :

```
List gabungan:
$Nama
[1] "Ali"      "Bambang" "Cici"
$Nilai
[1] 80 85 90
$Kelas
[1] A B A
Levels: A B
$Usia
[1] 20 21 19
$Jurusan
[1] "Matematika" "Fisika"      "Kimia"
$Status
[1] TRUE FALSE TRUE
```

Pembahasan : Program ini membuat dua list terpisah lalu menggabungkannya menjadi satu list besar menggunakan fungsi c(), sehingga seluruh elemen dari kedua list dapat diakses dalam satu struktur data terpadu.

Latihan

Latihan pemahaman tentang vector :

1. Buat vektor numerik berisi 6 skor ujian (misalnya: 70, 75, 80, 85, 90, 95).
2. Tambahkan 10 ke setiap elemen vektor menggunakan operasi vektorisasi.
3. Filter vektor untuk menampilkan skor di atas 80.
4. Buat vektor karakter berisi nama 5 siswa.

```
▶ # 1. Buat vektor numerik berisi 6 skor ujian
skor_ujian <- c(70, 75, 80, 85, 90, 95)
print("Vektor skor ujian asli:")
print(skor_ujian)

# 2. Tambahkan 10 ke setiap elemen vektor menggunakan operasi vektorisasi
skor_plus10 <- skor_ujian + 10
print("Setelah ditambah 10:")
print(skor_plus10)

# 3. Filter vektor untuk menampilkan skor di atas 80
skor_diatas80 <- skor_ujian[skor_ujian > 80]
print("Skor di atas 80:")
print(skor_diatas80)

# 4. Buat vektor karakter berisi nama 5 siswa
nama_siswa <- c("Andi", "Budi", "Citra", "Dewi", "Eka")
print("Nama siswa:")
print(nama_siswa)
```

output :

```
→ [1] "Vektor skor ujian asli:"
[1] 70 75 80 85 90 95
[1] "Setelah ditambah 10:"
[1] 80 85 90 95 100 105
[1] "Skor di atas 80:"
[1] 85 90 95
[1] "Nama siswa:"
[1] "Andi" "Budi" "Citra" "Dewi" "Eka"
```

Pembahasan : Program ini membuat vektor numerik berisi enam skor ujian, menambahkan nilai 10 ke setiap elemen menggunakan operasi vektorisasi, memfilter skor yang lebih dari 80, serta membuat vektor karakter berisi lima nama siswa untuk menunjukkan cara dasar manipulasi vektor numerik dan karakter di R.

Lakukan latihan berikut di Google Colab untuk mempraktikkan faktor:

- 1.Buat vektor karakter berisi 6 kota (misalnya: "Jakarta", "Bandung", "Surabaya", "Jakarta", "Medan", "Bandung").
- 2.Ubah menjadi faktor tidak terurut dan tampilkan level serta jumlah level.
- 3.Buat faktor terurut untuk tingkat kepuasan (misalnya: "Sangat Tidak Puas", "Tidak Puas", "Puas", "Sangat Puas") dengan 4 observasi.
- 4.Buat data frame dengan kolom Kota (faktor) dan Populasi (numerik).

```
▶ # 1. Buat vektor karakter berisi 6 kota
kota <- c("Jakarta", "Bandung", "Surabaya", "Jakarta", "Medan", "Bandung")
print("Vektor kota:")
print(kota)

# 2. Ubah menjadi faktor tidak terurut dan tampilkan level serta jumlah level
faktor_kota <- factor(kota)
print("Faktor kota (tidak terurut):")
print(faktor_kota)
print("Level faktor kota:")
print(levels(faktor_kota))
print(paste("Jumlah level:", nlevels(faktor_kota)))

# 3. Buat faktor terurut untuk tingkat kepuasan
tingkat_kepuasan <- factor(
  c("Puas", "Sangat Puas", "Tidak Puas", "Sangat Tidak Puas"),
  levels = c("Sangat Tidak Puas", "Tidak Puas", "Puas", "Sangat Puas"),
  ordered = TRUE
)
print("Faktor terurut untuk tingkat kepuasan:")
print(tingkat_kepuasan)

# 4. Buat data frame dengan kolom Kota (faktor) dan Populasi (numerik)
populasi <- c(10500000, 3500000, 2900000, 10500000, 2500000, 3500000)
data_kota <- data.frame(Kota = faktor_kota, Populasi = populasi)
print("Data frame Kota dan Populasi:")
print(data_kota)
```

Output :

```
☒ [1] "Vektor kota:"
[1] "Jakarta" "Bandung" "Surabaya" "Jakarta" "Medan"     "Bandung"
[1] "Faktor kota (tidak terurut):"
[1] Jakarta Bandung Surabaya Jakarta Medan     Bandung
Levels: Bandung Jakarta Medan Surabaya
[1] "Level faktor kota:"
[1] "Bandung" "Jakarta" "Medan"     "Surabaya"
[1] "Jumlah level: 4"
[1] "Faktor terurut untuk tingkat kepuasan:"
[1] Puas           Sangat Puas   Tidak Puas      Sangat Tidak Puas
Levels: Sangat Tidak Puas < Tidak Puas < Puas < Sangat Puas
[1] "Data frame Kota dan Populasi:"
  Kota Populasi
  1 Jakarta 10500000
  2 Bandung 3500000
  3 Surabaya 2900000
  4 Jakarta 10500000
  5 Medan 2500000
  6 Bandung 3500000
```

Pembahasan : Program ini membuat vektor karakter berisi enam nama kota, mengubahnya menjadi faktor tidak terurut untuk melihat level dan jumlahnya, membuat faktor terurut untuk tingkat kepuasan berdasarkan urutan logis, lalu membentuk data frame yang berisi kolom *Kota* (sebagai faktor) dan *Populasi* (sebagai data numerik) untuk menunjukkan penggunaan faktor dan data frame dalam R.

Lakukan latihan berikut di Google Colab untuk memperkuat pemahaman tentang data frame:

1. Buat data frame berisi 5 siswa dengan kolom Nama, Skor_Matematika, Skor_Fisika, dan Jurusan (faktor: IPA/IPS).
2. Tambahkan kolom Rata_rata yang menghitung rata-rata Skor_Matematika dan Skor_Fisika.
3. Filter siswa dengan Rata_rata di atas 80 menggunakan dplyr.

```
# --- 1. Buat data frame berisi 5 siswa ---
data_siswa <- data.frame(
  Nama = c("Andi", "Budi", "Citra", "Dewi", "Eka"),
  Skor_Matematika = c(85, 70, 90, 75, 88),
  Skor_Fisika = c(80, 65, 95, 78, 84),
  Jurusan = factor(c("IPA", "IPS", "IPA", "IPS", "IPA"))
)

print("Data awal:")
print(data_siswa)

# --- 2. Tambahkan kolom Rata_rata ---
data_siswa <- data_siswa %>%
  mutate(Rata_rata = (Skor_Matematika + Skor_Fisika) / 2)

print("Data dengan kolom Rata_rata:")
print(data_siswa)

# --- 3. Filter siswa dengan Rata_rata di atas 80 ---
siswa_diatas80 <- data_siswa %>%
  filter(Rata_rata > 80)
print("Siswa dengan Rata_rata di atas 80:")
print(siswa_diatas80)
```

output :

```
[1] "Data awal:"
  Nama Skor_Matematika Skor_Fisika Jurusan
1 Andi      85          80     IPA
2 Budi      70          65     IPS
3 Citra     90          95     IPA
4 Dewi      75          78     IPS
5 Eka       88          84     IPA
[1] "Data dengan kolom Rata_rata:"
  Nama Skor_Matematika Skor_Fisika Jurusan Rata_rata
1 Andi      85          80     IPA      82.5
2 Budi      70          65     IPS      67.5
3 Citra     90          95     IPA      92.5
4 Dewi      75          78     IPS      76.5
5 Eka       88          84     IPA      86.0
[1] "Siswa dengan Rata_rata di atas 80:"
  Nama Skor_Matematika Skor_Fisika Jurusan Rata_rata
1 Andi      85          80     IPA      82.5
2 Citra     90          95     IPA      92.5
3 Eka       88          84     IPA      86.0
```

Pembahasan : Program ini membuat data frame berisi data 5 siswa dengan nilai Matematika, Fisika, dan jurusan, kemudian menambahkan kolom **Rata_rata** menggunakan fungsi `mutate()` untuk menghitung nilai rata-rata dari dua mata pelajaran, lalu memfilter dan menampilkan hanya siswa yang memiliki **Rata_rata** lebih dari 80..

C. PEMBAHASAN TUGAS

Tugas

Lakukan latihan berikut di Google Colab untuk memperkuat pemahaman tentang matriks dan array:

1. Buat matriks 3x2 berisi angka 1 hingga 6, dengan pengisian per baris.
2. Tambahkan nama baris dan kolom ke matriks tersebut.
3. Hitung transpos dan rata-rata per kolom dari matriks.
4. Buat array 3D dengan dimensi 2x2x2 berisi angka 1 hingga 8.

```

▶ # 1. Buat matriks 3x2 berisi angka 1 hingga 6, dengan pengisian per baris
matriks <- matrix(1:6, nrow = 3, ncol = 2, byrow = TRUE)
print("Matriks 3x2 (diisi per baris):")
print(matriks)

# 2. Tambahkan nama baris dan kolom
rownames(matriks) <- c("Baris1", "Baris2", "Baris3")
colnames(matriks) <- c("Kolom1", "Kolom2")
print("Matriks dengan nama baris dan kolom:")
print(matriks)

# 3. Hitung transpos dan rata-rata per kolom
transpos <- t(matriks)
print("Transpos dari matriks:")
print(transpos)

rata_per_kolom <- colMeans(matriks)
print("Rata-rata per kolom:")
print(rata_per_kolom)

# 4. Buat array 3D dengan dimensi 2x2x2 berisi angka 1 hingga 8
array_3d <- array(1:8, dim = c(2, 2, 2))
print("Array 3D 2x2x2:")
print(array_3d)

```

output :

```

[1] "Matriks 3x2 (diisi per baris):"
[1] [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[1] "Matriks dengan nama baris dan kolom:"
      Kolom1 Kolom2
Baris1      1      2
Baris2      3      4
Baris3      5      6
[1] "Transpos dari matriks:"
      Baris1 Baris2 Baris3
Kolom1      1      3      5
Kolom2      2      4      6
[1] "Rata-rata per kolom:"
      Kolom1 Kolom2
                  3      4
[1] "Array 3D 2x2x2:"
[1] , , 1

      [,1] [,2]
[1,]    1    3
[2,]    2    4
      , , 2

      [,1] [,2]
[1,]    5    7
[2,]    6    8

```

Pembahasan : Program ini membuat matriks 3x2 berisi angka 1 hingga 6 yang diisi per baris dan diberi nama baris serta kolom, kemudian menghitung transpos serta rata-rata per kolom, dan selanjutnya membuat array 3 dimensi (2x2x2) berisi angka 1 hingga 8 untuk menunjukkan representasi data multidimensi di R.

Lakukan latihan berikut di Google Colab untuk memperkuat pemahaman tentang list:

1. Buat list berisi vektor Skor (numerik), Nama (karakter), dan Kategori (faktor).
2. Tambahkan elemen baru berupa matriks 2x2 ke list tersebut.
3. Akses dan ubah elemen kedua dari vektor Nama.

```
▶ # 1. Buat list berisi vektor Skor (numerik), Nama (karakter), dan Kategori (faktor)
Skor <- c(85, 90, 78, 92)
Nama <- c("Andi", "Budi", "Citra", "Dewi")
Kategori <- factor(c("A", "A", "B", "A"))

data_list <- list(Skor = Skor, Nama = Nama, Kategori = Kategori)
print("List awal:")
print(data_list)

# 2. Tambahkan elemen baru berupa matriks 2x2 ke list tersebut
matriks_baru <- matrix(1:4, nrow = 2, ncol = 2)
data_list$Matriks <- matriks_baru

print("List setelah ditambah elemen matriks:")
print(data_list)

# 3. Akses dan ubah elemen kedua dari vektor Nama
print("Elemen kedua dari Nama sebelum diubah:")
print(data_list$Nama[2])

data_list$Nama[2] <- "Bambang"

print("Elemen kedua dari Nama setelah diubah:")
print(data_list$Nama[2])

print("List akhir:")
print(data_list)
```

output :

```
→ [1] "List awal:"
$Skor
[1] 85 90 78 92

$Nama
[1] "Andi" "Budi" "Citra" "Dewi"

$Kategori
[1] A A B A
Levels: A B

[1] "List setelah ditambah elemen matriks:"
$Skor
[1] 85 90 78 92

$Nama
[1] "Andi" "Budi" "Citra" "Dewi"

$Kategori
[1] A A B A
Levels: A B

$Matriks
 [,1] [,2]
[1,]    1    3
[2,]    2    4

[1] "Elemen kedua dari Nama sebelum diubah:"
[1] "Budi"
[1] "Elemen kedua dari Nama setelah diubah:"
[1] "Bambang"
[1] "List akhir:"
$Skor
[1] 85 90 78 92

$Nama
[1] "Andi" "Bambang" "Citra" "Dewi"

$Kategori
[1] A A B A
Levels: A B

$Matriks
 [,1] [,2]
[1,]    1    3
[2,]    2    4
```

Pembahasan : Program ini membuat list yang berisi vektor numerik, karakter, dan faktor, lalu menambahkan elemen baru berupa matriks 2x2, kemudian mengakses dan mengubah elemen kedua dari vektor Nama, sehingga menghasilkan list akhir yang berisi data lengkap dengan elemen yang telah dimodifikasi.

D. KESIMPULAN

Kesimpulan dari hasil praktikum *Statistika Modeling Pemrograman R menggunakan Google Colab* adalah bahwa Google Colab dapat digunakan sebagai platform efektif untuk menjalankan bahasa pemrograman R dalam melakukan analisis data, mulai dari membuat dan memanipulasi vektor, faktor, data frame, matriks, array, hingga list, serta menerapkan operasi aritmatika, logika, pengelompokan data, dan visualisasi sederhana, sehingga memudahkan proses *data analysis* dan *statistical modeling* secara interaktif dan efisien tanpa perlu instalasi tambahan di perangkat lokal.