```
print(R.version.string )
rm(list = ls())
print(2 + 3 )
print(10 – 4)
print( 6 * 5 )
print( 20 / 4 )
print( 2^5)
v <- c(1, 2, 3, 4, 5)
print(v)
print( getwd() )
print( ls())
rm(v)
print( ls())
help(mean)
?mean
library(ggplot2)
 q()
```

```
library(dplyr)
data < - data.frame(
  x1 = 1:6,
  x2 = c(1, 2, 2, 3, 1, 2),
  x3 = c("F", "B", "C", "E", "A", "D")
)
print(“Original Data ”)
print(data)
arranged_data < - arrange(data, x2)
print("Arranged by x2:")
print(arranged_data)
filtered_data < - filter(data, x2 == 2)
print("Filtered where x2 == 2:")
print(filtered_data)
mutated_data < - mutate(data, x4 = x2 * 2)
print("After mutation (x4 = x2 * 2):")
print(mutated_data)
x3_values < - pull(data, x3)
print("Pulled x3 column:")
print(x3_values)
renamed_data < - rename(data, ID = x1)
print("Renamed x1 to ID:")
print(renamed_data)
sampled_data < - sample_n(data, 3)
print("Random sample of 3 rows:")
print(sampled_data)
selected_data < - select(data, x1, x3)
print("Selected x1 and x3 columns:")
print(selected_data)
```

```
data("swiss")
selected_rows < - c(1, 2, 3, 10, 11, 12, 13)
selected_columns < - c("Examination", "Education", "Infant.Mortality")
swiss_subset < - swiss[selected_rows, selected_columns]
swiss_subset[4, "Infant.Mortality"] < - NA
total_row < - colSums(swiss_subset, na.rm = TRUE)
swiss_subset < - rbind(swiss_subset, Total = total_row)
swiss_subset$Examination_Proportion < - round(swiss_subset$Examination / total_row["Examina
3)
print(swiss_subset)
```

```
library(dplyr)
hotel_data < - data.frame(
  Booking_ID = paste0("BKG", 101:110),
  Booking_Date = as.Date(c("2025 -07-01", "2025 -07-03", "2025 -07-05", "2025 -07-06", "202
                           "2025 -07-08", "2025 -07-09", "2025 -07-10", "2025 -07-11", "202
  Children = c(1, 0, 2, 1, 0, 2, 0, 1, 2, 0),
  Adults = c(2, 1, 2, 2, 1, 2, 1, 2, 2, 1),
  Length_of_Stay = c(3, 2, 5, 1, 2, 4, 2, 3, 6, 2),
  Parking_Spaces = c(1, 0, 1, 0, 1, 1, 0, 1, 1, 0),
  Cancellations = c(0, 1, 0, 0, 1, 0, 1, 0, 0, 1)
)
print("Original Hotel Booking Data:")
print(hotel_data)
filtered_data < - filter(hotel_data, Children > 1)
print("Filtered Bookings (more than 1 child):")
print(filtered_data)
mutated_data < - mutate(hotel_data, Total_Guests = Children + Adults)
print("Data with Total_Guests column:")
print(mutated_data)
stay_lengths < - pull(hotel_data, Length_of_Stay)
print("Length of Stay column as vector:")
print(stay_lengths)
renamed_data < - rename(hotel_data, Parking_Spots = Parking_Spaces)
print("Renamed Data (Parking_Spaces to Parking_Spots):")
print(renamed_data)
```

```
dates < - as.Date(c("2023 -Mar-01", "2023 -Apr-15", "2023 -May-10", "2023 -Jun-25", "2023 -
format="%Y -%b-%d")
cat("Dates: \n")
print(dates)
names < - c("Alice", "Bob", "Charlie", "Diana", "Eve")
scores < - c(85, 92, 78, 88, 95)
combined1 < - paste(names, "visited on", dates)
combined2 < - paste(names, "scored", scores, "on", dates)
cat(" \nCombined Name + Date: \n")
print(combined1)
cat(" \nCombined Name + Score + Date: \n")
print(combined2)
```

```
students < - read.csv( “C:\\Users \\system10 \\Documents \\R Studio.program \\students.csv
stringsAsFactors = FALSE)
cat("Size of Data Frame: \n")
cat("Number of Rows:", nrow(students), " \n")
cat("Number of Columns:", ncol(students), " \n\n")
cat("Class of Each Column: \n")
print(sapply(students, class))
cat(" \nColumn Names: \n")
print(colnames(students))
cat(" \nRow Names: \n")
print(rownames(students))
cat(" \nFirst 3 Rows: \n")
print(head(students, 3))
cat(" \nLast 2 Rows: \n")
print(tail(students, 2))
cat(" \nAccess single value [2,3] (2nd row, 3rd column): \n")
print(students[2, 3])
cat(" \nAccess entire 1st row: \n")
print(students[1, ])
cat(" \nAccess entire 'Marks' column: \n")
print(students$Marks)
students$Result < - ifelse(students$Marks >= 75, "Pass", "Fail")
students$Result < - as.factor(students$Result)
students$Grade < - ifelse(students$Marks >= 85, "Distinction",
                          ifelse(students$Marks >= 75, "First Class", "Second Class"))
students$Grade < - as.factor(students$Grade)
new_student < - data.frame(
  Name = "Frank",
  Age = 22,
  Gender = "Male",
  Department = "Maths",
  Marks = 78,
  Result = factor("Pass", levels = levels(students$Result)),
  Grade = factor("First Class", levels = levels(students$Grade))
)
students < - rbind(students, new_student)
students$Gender < - as.factor(students$Gender)
students$Department < - as.factor(students$Department)
cat(" \nFactor Levels: \n")
cat("Gender:", levels(students$Gender), " \n")
cat("Department:", levels(students$Department), " \n")
cat("Grade:", levels(students$Grade), " \n")
cat(" \nSummary of Data Frame: \n")
print(summary(students))
cat(" \nFrequency Tables: \n")
cat("Gender: \n")
print(table(students$Gender))
cat("Department: \n")
print(table(students$Department))
cat("Grade: \n")
print(table(students$Grade))
cat("Result: \n")
print(table(students$Result))
NOTE:
Name,Age,Gender,Department,Marks
Alice,21,Female,Maths,88
Bob,22,Male,Physics,75
Charlie,20,Male,Chemistry,95
Diana,21,Female,Maths,82
Evan,23,Male,Physics,70
```

```
x <- c(3, 1, 4, 1, 5, 9)
print( x[1] )         # First element
print( x[c(2, 4, 6)] ) # Elements at positions 2, 4, and 6
print( x[-1] )        # Exclude the first element
print( x[-c(2, 5)] )   # Exclude elements at positions 2 and 5
print( x[0])          # Returns an empty vector
print( x[c(TRUE, FALSE, TRUE, FALSE, TRUE, FALSE)] ) # Picks 1st, 3rd, 5th elements
print( x[])           # Returns all elements of x
print( x[c(1, 3, 5)] )  # Valid mix of positive indices
print( x[c(TRUE, TRUE, FALSE, TRUE, FALSE, TRUE)] )  # Valid boolean indexing
```

```
raw_names < - c(" alice smith ", "BOB JOHNSON", "charlie Brown", " diana LEE", "EV AN kelly
clean_names < - trimws(raw_names)
lower_names < - tolower(clean_names)
formatted_names < - tools::toTitleCase(lower_names)
split_names < - strsplit(formatted_names, " ")
first_names < - sapply(split_names, `[`, 1)
last_names < - sapply(split_names, `[`, 2)
students_df < - data.frame(
 FullName = formatted_names,
 FirstName = first_names,
 LastName = last_names,
 stringsAsFactors = FALSE
)
print("Cleaned Student Name Data:")
print(students_df)
```

```
student_names < - c("Anita", "Bala", "Charan", "Divya", "Elango",
 "Farah", "Ganesh", "Hema", "Imran", "Jaya")
marks < - c(78, 85, 67, 90, 76, 85, 92, 88, 70, 80)
avg < - mean(marks)
count < - length(marks)
first < - marks[1]
last < - marks[length(marks)]
maximum < - max(marks)
med < - median(marks)
minimum < - min(marks)
get_mode < - function(x) {
 ux <- unique(x)
 ux[which.max(tabulate(match(x, ux)))]
}
mode_val < - get_mode(marks)
rms < - sqrt(mean(marks^2))
stddev < - sd(marks)
total < - sum(marks)
cat("Student Marks Aggregation Results: \n")
cat(" ---------------------------------- \n")
cat("Average (Mean):", avg, " \n")
cat("Count:", count, " \n")
cat("First Mark (", student_names[1], "):", first, " \n")
cat("Last Mark (", student_names[10], "):", last, " \n")
cat("Maximum:", maximum, " \n")
cat("Median:", med, " \n")
cat("Minimum:", minimum, " \n")
cat("Mode:", mode_val, " \n")
cat("RMS (Root Mean Square):", rms, " \n")
cat("Standard Deviation:", stddev, " \n")
cat("Sum:", total, " \n")
```

```
if (!require("reshape2")) install.packages("reshape2", dependencies = TRUE)
library(reshape2)
names < - c("Anita", "Bala", "Charan", "Divya", "Elango",
           "Farah", "Ganesh", "Hema", "Imran", "Jaya")
addresses < - c("Chennai", "Madurai", "Coimbatore", "Trichy", "Salem",
                "Erode", "Tirunelveli", "Vellore", "Thanjavur", "Nagercoil")
states < - c("Tamil Nadu", "Tamil Nadu", "Tamil Nadu", "Tamil Nadu", "Tamil Nadu",
             "Tamil Nadu", "Tamil Nadu", "Tamil Nadu", "Tamil Nadu", "Tamil Nadu")
df_cbind < - cbind(Name = names, Address = addresses, State = states)
cat("cbind() Output: \n")
print(df_cbind)
row1 < - c("Anita", "Chennai", "Tamil Nadu")
row2 < - c("Bala", "Madurai", "Tamil Nadu")
row3 < - c("Charan", "Coimbatore", "Tamil Nadu")
df_rbind < - rbind(row1, row2, row3)
colnames(df_rbind) < - c("Name", "Address", "State")
cat(" \nrbind() Output (3 rows): \n")
print(df_rbind)
student_df < - data.frame(
  ID = 1:10,
  Name = names,
  Address = addresses,
  State = states,
  stringsAsFactors = FALSE
)
melted_df < - melt(student_df, id.vars = "ID")
cat(" \nMelted Data: \n")
print(melted_df)
casted_df < - dcast(melted_df, ID ~ variable)
cat(" \nCasted (reshaped) Data: \n")
print(casted_df)
```

```
customers < - c(45, 52, 48, 60, 55, 49, 62, 58, 53, 47,
               51, 50, 65, 59, 54, 56, 61, 57, 63, 46)
print(mean(customers))   # Average number of customers per day
print(sd(customers))     # Standard deviation of customers
print(pnorm(55, mean=mean(customers), sd=sd(customers)))
print(pnorm(60, mean=mean(customers), sd=sd(customers)) -
         pnorm(50, mean=mean(customers), sd=sd(customers)))
hist(customers, probability=TRUE, col="lightgreen",
     main="Customer Arrivals Distribution",
     xlab="Number of Customers")
curve(dnorm(x, mean=mean(customers), sd=sd(customers)),
      col="red", lwd=2, add=TRUE)
print(dbinom(7, size=10, prob=0.6))     # Probability exactly 7 buy coffee
print(pbinom(7, size=10, prob=0.6))    # Probability ≤ 7 buy coffee
print(dpois(5, lambda=3))               # Probability of exactly 5 complaints
print(ppois(5, lambda=3))               # Probability of ≤ 5 complaints
```

```
n <- 12
p <- 1/5
prob_4_or_less < - pbinom(4, size = n, prob = p)
cat("Probability of getting 4 or fewer correct answers:", prob_4_or_less, " \n")
mean_val < - n * p
variance_val < - n * p * (1 - p)
sd_val < - sqrt(variance_val)
cat(" \nDescriptive Statistics: \n")
cat("Mean:", mean_val, " \n")
cat("Variance:", variance_val, " \n")
cat("Standard Deviation:", sd_val, " \n")
x <- 0:n
probs < - dbinom(x, size = n, prob = p)
prob_table < - data.frame(Correct_Answers = x, Probability = probs)
print(prob_table)
barplot(probs,
        names.arg = x,
        col = "skyblue",
        main = "Binomial Distribution: P(X = x) for 12 MCQs (p = 0.2)",
        xlab = "Number of Correct Answers",
        ylab = "Probability")
abline(v = 4.5, col = "red", lty = 2)
legend("topright", legend = c("X ≤ 4"), col = "red", lty = 2)
```

```
cat(" \n--- ONE -SAMPLE t -TEST: Energy Bar Weights ---\n")
energy_bars < - c(24, 26, 23, 27, 25, 28, 22, 24, 26, 25) # sample data
t_one < - t.test(energy_bars, mu = 25) # mu = claimed population mean
print(t_one) # prints t -test results
if(t_one$p.value < 0.05) {
  cat("Conclusion: Average weight is significantly different from 25g. \n")
} else {
  cat("Conclusion: No evidence that average weight differs from 25g. \n")
}
cat(" \n--- TWO -SAMPLE t -TEST: Fertilizer Effect on Plant Height ---\n")
fertilizer_A < - c(14, 15, 16, 14, 15, 17, 16) # heights in cm
fertilizer_B < - c(18, 19, 17, 20, 18, 19, 21) # heights in cm
t_two < - t.test(fertilizer_A, fertilizer_B, var.equal = TRUE)
print(t_two)
if(t_two$p.value < 0.05) {
  cat("Conclusion: Fertilizer type significantly affects plant height. \n")
} else {
  cat("Conclusion: No significant difference in plant height between fertilizers. \n")
}
cat(" \n--- PAIRED t -TEST: Fitness Training Push -Up Improvement ---\n")
pushups_before < - c(56, 49, 53, 52, 50, 48, 51) # before training
pushups_after  < - c(58, 50, 55, 53, 51, 49, 54) # after training
t_paired < - t.test(pushups_before, pushups_after, paired = TRUE)
print(t_paired)
if(t_paired$p.value < 0.05) {
  cat("Conclusion: Training program significantly improved push -up counts. \n")
} else {
  cat("Conclusion: No significant improvement in push -up counts. \n")
}
```

```
X <- c(10, 20, 30, 40, 50, 60, 70, 80, 90, 100)  # e.g., study hours
Y <- c(15, 22, 29, 35, 45, 52, 60, 72, 81, 95)   # e.g., test scores
correlation < - cor(X, Y)
cat("Correlation between X and Y:", correlation, " \n")
reg_model < - lm(Y ~ X)
cat(" \nRegression Summary: \n")
print(summary(reg_model))
plot(X, Y ,
     main = "Regression and Correlation",
     xlab = "X (Independent Variable)",
     ylab = "Y (Dependent Variable)",
     col = "blue", pch = 16)
abline(reg_model, col = "red", lwd = 2)
legend("topleft", legend = c("Data Points", "Regression Line"),
       col = c("blue", "red"), pch = c(16, NA), lty = c(NA, 1))
```

```
traditional < - c(23, 18, 30, 15, 27)       # Traditional lectures
discussion  < - c(21, 25, 20, 28, 19)       # Group discussion
online      < - c(30, 26, 33, 24, 29)       # Online learning
scores < - c(traditional, discussion, online)
method < - factor(c(rep("Traditional Lecture", length(traditional)),
                    rep("Group Discussion", length(discussion)),
                    rep("Online Learning", length(online))))
data < - data.frame(scores, method)
cat(" \n--- One-way ANOV A Results ---\n")
anova_result < - aov(scores ~ method, data = data)
print(summary(anova_result))
cat(" \n--- Kruskal -Wallis Test Results ---\n")
kruskal_result < - kruskal.test(scores ~ method, data = data)
print(kruskal_result)
boxplot(scores ~ method, data = data,
        main = "Exam Scores by Teaching Method",
        xlab = "Teaching Method",
        ylab = "Exam Score",
        col = c("skyblue", "lightgreen", "lightpink"))
```

```
students < - read.csv( “C:\\Users \\system10 \\Documents \\R Studio.program \\students.csv
print("Data Frame Content:")
print(students)
print("Number of rows and columns:")
print(dim(students))      # size of data frame (rows, columns)
print("Class of each column:")
print(sapply(students, class))   # data type of each column
print("Row names:")
print(rownames(students))        # names of rows
print("Column names:")
print(colnames(students))        # names of columns
print("First 3 rows:")
print(head(students, 3))         # first 3 rows
print("Last 2 rows:")
print(tail(students, 2))         # last 2 rows
```

NOTE:

Let’s assume you have this CSV file saved as students.csv in your working directory:

```
ID Name  Age Gender  Score
1 Alice  20 F 85
2 Bob 22 M 78
3 Charlie  21 M 92
4 Diana  23 F 88
5 Ethan  20 M 75
```

```
students < - read.csv( “C:\\Users \\system10 \\Documents \\R Studio.program \\students.csv
print("Access 2nd row, 3rd column (Age of 2nd student):")
print(students[2, 3])
print("All Names column:")
print(students$Name)
students$Result < - ifelse(students$Score >= 80, "Pass", "Fail")
students$Result< -as.factor(students$Result)
print("Data after adding Result column:")
print(students)
new_row < - data.frame(ID = 6, Name = "Fiona", Age = 22, Gender = "F", Score = 90, Result =
students < - rbind(students, new_row)
print("Data after adding new row:")
print(students)
students$Gender < - as.factor(students$Gender)  # ensure Gender is factor
print("Levels of Gender factor:")
print(levels(students$Gender))
print("Summary of data frame:")
print(summary(students))
```

```
library(pwr)
alpha < - 0.05
power < - 0.80
mean_null< - 20000
mean_trial< - 14500
sd_trial< - 6000
effect_size_income< - (mean_trial - mean_null) / sd_trial
print(paste("Effect size for income test:", round(effect_size_income, 3)))
sample_size_income< - pwr.t.test(d = effect_size_income, sig.level = alpha,
                    power = power, type = "one.sample", alternative = "less")
print("Sample size calculation for income test:")
print(sample_size_income)
sleep_change< - c(1.2, -0.5, 0.8, -1.0, 0.6, -0.3, 0.9, -1.2, 0.4, -0.8)
mean_sleep< - mean(sleep_change)
sd_sleep< - sd(sleep_change)
effect_size_sleep< - mean_sleep / sd_sleep
print(paste("Effect size for sleep change test:", round(effect_size_sleep, 3)))
sample_size_sleep< - pwr.t.test(d = effect_size_sleep,
sig.level = alpha, power = power,type = "one.sample", alternative = "two.sided")
print("Sample size calculation for sleep change test:")
print(sample_size_sleep)
```

```
library(pwr)
alpha < - 0.05
power < - 0.80
mean_men_cal< - 2350.2
sd_men_cal< - 258
mean_women_cal< - 1872.4
sd_women_cal< - 420
pooled_sd_cal< - sqrt(((sd_men_cal^2) + (sd_women_cal^2)) / 2)
effect_size_cal< - (mean_men_cal - mean_women_cal) / pooled_sd_cal
cat("Effect size (Calories):", round(effect_size_cal, 3), " \n")
sample_size_cal< - pwr.t.test(d = effect_size_cal,  sig.level = alpha,  power = power,
                              type = "two.sample",  alternative = "two.sided")
cat("Sample size (Calories): \n")
print(sample_size_cal)
protein_men< - c(7.1, 6.8, 7.4, 7.0, 6.9)
protein_women< - c(6.5, 6.7, 6.8, 6.4, 6.6)
mean_men_protein< - mean(protein_men)
mean_women_protein< - mean(protein_women)
sd_men_protein< - sd(protein_men)
sd_women_protein< - sd(protein_women)
pooled_sd_protein< - sqrt(((sd_men_protein^2) + (sd_women_protein^2)) / 2)
effect_size_protein< - (mean_men_protein - mean_women_protein) / pooled_sd_protein
cat("Effect size (Protein):", round(effect_size_protein, 3), " \n")
sample_size_protein< - pwr.t.test(d = effect_size_protein,
sig.level = alpha, power = power, type = "two.sample", alternative = "two.sided")
cat("Sample size (Protein): \n")
print(sample_size_protein)
glucose_men< - c(90, 85, 92, 88, 86)
glucose_women< - c(95, 96, 94, 98, 93)
mean_men_glucose< - mean(glucose_men)
mean_women_glucose< - mean(glucose_women)
sd_men_glucose< - sd(glucose_men)
sd_women_glucose< - sd(glucose_women)
pooled_sd_glucose< - sqrt(((sd_men_glucose^2) + (sd_women_glucose^2)) / 2)
effect_size_glucose< - (mean_men_glucose - mean_women_glucose) / pooled_sd_glucose
cat("Effect size (Glucose):", round(effect_size_glucose, 3), " \n")
sample_size_glucose< - pwr.t.test(d = effect_size_glucose,
sig.level = alpha,
                                  power = power,
                                  type = "two.sample",
                                  alternative = "less") # one -tailed
cat("Sample size (Glucose): \n")
print(sample_size_glucose)
```

```
library(dplyr)
library(tidyr)
library(stringr)
library(lubridate)
employees < - read.csv("C: \\Users \\system10 \\Documents \\R
Studio.program \\empolyees.csv",stringsAsFactors = FALSE)
print("1.1head employees")
print(head(employees))
print("1.2number of rows and columns")
print(dim(employees)) # Number of rows & columns
print(" 2.1structure of data frame")
str(employees)# Structure of data frame
print(" 2.2column names")
print(names(employees))         # Column names
print("3.Filtering: Employees in IT with salary > 70000")
high_paid_IT< - employees %>%
  filter(Department == "IT", Salary > 70000)
print(high_paid_IT)
print("4.Selecting columns: Name, Department, Salary")
selected_cols< - employees %>%
  select(Name, Department, Salary)
print(selected_cols)
print("5.Adding new column: Annual Bonus (10% of Salary)")
employees < - employees %>%
  mutate(AnnualBonus = Salary * 0.10)
print(employees)
print("6.Grouping: Average salary by department")
avg_salary_dept< - employees %>%
  group_by(Department) %>%
  summarise(AvgSalary = mean(Salary), Count = n(), .groups = "drop")
print(avg_salary_dept)
print("7.String handling: Extract FirstName and LastName")
employees < - employees %>%
  mutate(
    FirstName = word(Name, 1),
    LastName = word(Name, 2)
  )
print(employees[, c("Name","FirstName","LastName")])
print("8.Extract email domain")
employees < - employees %>%
  mutate(EmailDomain = str_extract(Email, "(?<=@).*"))
print(employees[, c("Name","EmailDomain")])
print("9. Categorical variable: Salary category")
employees < - employees %>%
  mutate(SalaryCategory = case_when(
    Salary >= 75000 ~ "High",
    Salary >= 55000 ~ "Medium",
    TRUE ~ "Low"
  ))
print(employees[, c("Name","Salary","SalaryCategory")])
print("10.Reshaping: Long format for Salary and Bonus")
long_format< - employees %>%
  pivot_longer(cols = c(Salary, AnnualBonus),
               names_to = "Metric", values_to = "Value")
print(long_format)
print("11.Merge with department locations dataset")
dept_locations< - data.frame(
  Department = c("HR", "IT", "Finance"),
  Location = c("New York", "Bangalore", "Mumbai")
)
print(dept_locations)
print("11.1employees merged ")
```

```
employees_merged< - left_join(employees, dept_locations, by = "Department")
print(employees_merged)
print("12.Summary statistics")
summary_stats< - employees %>%
  summarise(
    MinSalary = min(Salary),
    MaxSalary = max(Salary),
    AvgSalary = mean(Salary),
    SD_Salary = sd(Salary)
  )
print(summary_stats)
```

```
car_data < - data.frame(
  mpg  = c(21, 22, 18, 24, 27, 16, 19, 23, 15, 26),
  wt   = c(2.620, 2.875, 3.215, 2.200, 1.835, 3.460, 3.050, 2.780, 3.840, 1.935),
  disp = c(160, 160, 225, 146, 108, 258, 225, 140, 360, 105),
  hp   = c(110, 110, 105, 95, 93, 110, 123, 102, 245, 91)
)
model < - lm(mpg ~ wt + disp + hp, data = car_data)
print( summary(model) )
new_car < - data.frame(wt = 2.5, disp = 150, hp = 100)
predicted_mpg < - predict(model, newdata = new_car)
print( predicted_mpg )
pred_values < - predict(model)
plot(car_data$mpg, pred_values,
     xlab = "Actual MPG", ylab = "Predicted MPG",
     main = "Actual vs Predicted MPG",
     pch = 19, col = "blue")
abline(a=0, b=1, col="red") # perfect prediction line
```

```
ggplot2"))
library(tidyverse)
library(broom)
library(rsample)
library(yardstick)
library(car)
library(lmtest)
library(sandwich)
library(performance)
library(ggplot2)
insurance < - read_csv( “C:\\Users \\system10 \\Documents \\R Studio.program \\insurance.cs
show_col_types = FALSE)
if ("sex" %in% names(insurance) && !"gender" %in% names(insurance)) {
insurance < - insurance |> rename(gender = sex)
}
needed < - c("age","gender","bmi","smoker","region","charges")
stopifnot(all(needed %in% names(insurance)))
insurance < - insurance |> select(all_of(needed))
insurance < - insurance |>
 mutate(
    gender = factor(gender),
    smoker = factor(smoker),   # expects levels like "yes"/"no" or "Yes"/"No"
    region = factor(region)
  )
glimpse(insurance)
summary(insurance)
set.seed(123)
split < - initial_split(insurance, prop = 0.8, strata = smoker)
train < - training(split)
test  < - testing(split)
m1 < - lm(charges ~ age, data = train)
m2 < - lm(charges ~ age + gender + bmi + smoker + region, data = train)
m3 < - lm(charges ~ age + gender + bmi + smoker + region +
           bmi:smoker + age:smoker, data = train)
m4 < - lm(log(charges) ~ age + gender + bmi + smoker + region +
           bmi:smoker, data = train)
cat(" \n================== \nMODEL 1: charges ~ age \n================== \n")
print(summary(m1))
cat(" \n================== \nMODEL 2: charges ~ age + gender + bmi + smoker +
region \n================== \n")
print(summary(m2))
cat(" \n[Robust SEs (HC3) for M2] \n")
print(coeftest(m2, vcov = sandwich::vcovHC(m2, type = "HC3")))
cat(" \n================== \nMODEL 3: + interactions (bmi:smoker,
age:smoker) \n================== \n")
print(summary(m3))
cat(" \n[Robust SEs (HC3) for M3] \n")
print(coeftest(m3, vcov = sandwich::vcovHC(m3, type = "HC3")))
cat(" \n================== \nMODEL 4 (log): log(charges) ~ ... \n================== \n")
print(summary(m4))
cat(" \n--- VIF (M2) ---\n"); print(vif(m2))
cat(" \n--- VIF (M3) ---\n"); print(vif(m3))
cat(" \n--- Breusch -Pagan test for heteroskedasticity ---\n")
cat("M2: \n"); print(bptest(m2))
cat("M3: \n"); print(bptest(m3))
cat("M4 (on log scale): \n"); print(bptest(m4))
eval_model < - function(model, newdata, response = "charges", log_model = FALSE) {
  if (log_model) {
    pred_log < - predict(model, newdata = newdata)
    smear < - mean(exp(residuals(model)))
    pred < - exp(pred_log) * smear
  } else {
```

```
    pred < - predict(model, newdata = newdata)
  }
  tibble(
    truth = newdata[[response]],
    .pred = as.numeric(pred)
  ) |>
    metrics(truth = truth, estimate = .pred)
}
perf_m1 < - eval_model(m1, test)
perf_m2 < - eval_model(m2, test)
perf_m3 < - eval_model(m3, test)
perf_m4 < - eval_model(m4, test, log_model = TRUE)
cat(" \n=== Test -set Metrics (RMSE / MAE / R -squared) === \n")
print(perf_m1)
print(perf_m2)
print(perf_m3)
print(perf_m4)
plot_pred < - function(model, data, title, log_model = FALSE) {
  if (log_model) {
    pred_log < - predict(model, newdata = data)
    smear < - mean(exp(residuals(model)))
    data$.pred < - exp(pred_log) * smear
  } else {
    data$.pred < - predict(model, newdata = data)
  }
  ggplot(data, aes(x = .pred, y = charges)) +
    geom_point(alpha = 0.5) +
    geom_abline(intercept = 0, slope = 1, linewidth = 0.8) +
    labs(x = "Predicted charges", y = "Observed charges", title = title)
}
tidy_m2 < - tidy(m2, conf.int = TRUE)
tidy_m3 < - tidy(m3, conf.int = TRUE)
tidy_m4 < - tidy(m4, conf.int = TRUE)
cat(" \n--- Tidy Coefficients (M3) ---\n")
print(tidy_m3)
cat(" \nDone. \n")
Sample Insurance Data (CSV)
Here’s an example of what insurance .csv might look like:
63 male  29.4 no southwest  29663.59
20 male  36.2 no southeast  16976.24
46 female  19.2 no northwest  17178.83
52 male  32.5 no southwest  24986.95
56 female  28.0 no northeast  20922.97
35 male  21.3 no northeast  16229.48
37 male  27.8 yes southeast  49793.06
60 male  32.9 no southwest  28339.84
40 female  24.6 no northeast  18378.31
51 female  36.6 no northwest  26363.28
```