```
1 >  products <- list(
list(name = "Apple", price =120),
list(name = "Banana", price = 35),
list(name = "Milk", price = 25),
list(name = "Bread", price = 50),
list(name = "Eggs", price = 10))
shopping_cart <- list()
cart_items_to_add <- list(
list(name = "Apple", quantity = 3),
list(name = "Milk", quantity = 2))
for (item in cart_items_to_add) {
product_name <- item$name
quantity <- item$quantity
product <- NULL
for (p in products) {
if (p$name == product_name) {
product <- p
break}}
if (!is.null(product)) {
cart_item <- list(name = product$name, price = product$price, quantity =
quantity)
shopping_cart <- c(shopping_cart, list(cart_item))
cat("Item added to cart.\n")} else {
cat("Product not found.\n")}}
subtotal <- 0
cat("\nReceipt:\n")
for (item in shopping_cart) {
item_subtotal <- item$price * item$quantity
cat(sprintf("%s (%d units) - Price: $%.2f - Subtotal: $%.2f\n", item$name,
item$quantity, item$price, item_subtotal))
subtotal <- subtotal + item_subtotal}
tax_rate <- 0.08
tax_amount <- subtotal * tax_rate
total_cost_before_tax <- subtotal
total_cost <- total_cost_before_tax + tax_amount
cat("\nSubtotal: $%.2f\n", subtotal)
cat("Tax Amount (8%): $%.2f\n", tax_amount)
cat("Total Cost: $%.2f\n", total_cost)
```

```
2 > num_students <- 10
num_courses <- 5
student_names <- c("John", "Anna", "Tim", "Harry", "Pal", "Jim",
"Peter", "Bob", "Cook", "James")
course_marks <- matrix(c(
85, 92, 78, 88, 95, 88, 89, 78, 77, 81,
75, 80, 85, 70, 60, 90, 67, 70, 89, 87,
100, 78, 56, 34, 56,45, 78, 97, 66, 89,
78, 45, 67, 89, 90, 56, 89, 67, 99, 98,
89, 80, 67, 78, 90, 67, 78, 90, 78, 78),
 nrow = num_students, byrow = TRUE)
student_records <- list()
for (student_index in 1:num_students)
{ student_name <- student_names[student_index]
marks <- course_marks[student_index, ]
total_marks <- sum(marks)
average_marks <- total_marks / num_courses
grade <- ifelse(average_marks >= 90, "A", ifelse(average_marks >=
80,"B",
ifelse(average_marks >= 70, "C",
ifelse(average_marks >= 60, "D", "F"))))
student_record <- list(
name = student_name,
marks = marks,
total = total_marks,
average = average_marks,
grade = grade )
student_records <- c(student_records, list(student_record)) }
cat("\nStudent Grade Report:\n")
for (student_record in student_records) {
cat("\nName:", student_record$name, "\n")
cat("Marks:", student_record$marks, "\n")
cat("Total Marks:", student_record$total, "\n")
cat("Average Marks:", student_record$average, "\n")
cat("Grade:", student_record$grade, "\n")
```

```
3> fine_per_day = 2
cap_fine = 100
no_fine_days = 7
fine_days = 30
famt = 0
days_over = as.integer(readline("enter"))
if(days_over <= no_fine_days) {
  famt = 0
  cat("No fine :)")
} else if (days_over <= fine_days) {
  famt = (days_over - no_fine_days) *
fine_per_day
  cat("Pay fine ", famt)
} else {
  famt = cap_fine
  cat("Cap exceeded", famt)
}
```

```
4 > inv_items = character()
inv_qty = numeric()
store = function(item, qty) {
  inv_items <<- c(inv_items, item)
  inv_qty <<- c(inv_qty, qty)
  cat("Added\n")}
update = function(item, new_qty) {
  if (item %in% inv_items) {
    ind = which(inv_items == item)
    inv_qty[ind] <<- new_qty } else {
    cat("Item not found.\n")
    choice = as.character(readline("Y to add newly\n"))
    if (choice == 'y'){ store(item, new_qty)}}}
disp = function() {
  for(i in 1:length(inv_items)) {
    cat("\nItem: ", inv_items[i], "    Qty: ", inv_qty[i]) }
  print("\n")}
del = function(item) {
  if (item %in% inv_items) {
    ind = which(inv_items == item)
    inv_items <<- inv_items[-ind]
    inv_qty <<- inv_qty[-ind]
    cat("\nDeleted\n") } else {
    cat("\nItem not found\n")}}
while (TRUE) { ch = as.integer(readline("\nEnter choice: "))
  if (ch == 1) {
    item = as.character(readline("\nEnter item name: "))
    qty = as.numeric(readline("\nEnter qty: "))
    store(item, qty)  } else if (ch == 2) {
    item = as.character(readline("\nEnter item name: "))
    new_qty = as.numeric(readline("\nEnter new qty: "))
    update(item, new_qty)} else if (ch == 3) {
    disp() } else if (ch == 4) {
    item = as.character(readline("\nEnter item name to delete: "))
    del(item)}}
```

```
5> library(dplyr)
student_data <- data.frame(
  Name = character(), Math_Score = numeric(),
  Science_Score = numeric(),  History_Score = numeric(),
  Attendance = numeric())
add_student <- function(name, math_score, science_score, history_score,
attendance) {
  new_student <- data.frame(Name = name, Math_Score = math_score,
Science_Score = science_score,
                            History_Score = history_score, Attendance = attendance)
  student_data <<- bind_rows(student_data, new_student)
  cat("Student added.\n")}
calculate_avg_scores <- function() {
  student_data %>%
    mutate(Average_Score = (Math_Score + Science_Score + History_Score) / 3)
%>%
    select(Name, Average_Score)}
identify_low_attendance <- function(threshold) {
  student_data %>% filter(Attendance < threshold) %>%
    select(Name, Attendance)} generate_report <- function() {
  avg_scores <- calculate_avg_scores()
  low_attendance <- identify_low_attendance(70)
  report <- merge(avg_scores, low_attendance, by = "Name", all = TRUE)
  report$Attendance[is.na(report$Attendance)] <- 100  # Fill NA attendance
with 100%
  cat("Performance Report:\n")
  print(report)} while (TRUE) {
  cat("\n1. Add Student\n2. Generate Report\n3. Exit\n")
  choice <- as.integer(readline("Enter choice: "))
  if (choice == 1) { name <- readline("Enter student name: ")
    math_score <- as.numeric(readline("Enter math score: "))
    science_score <- as.numeric(readline("Enter science score: "))
    history_score <- as.numeric(readline("Enter history score: "))
    attendance <- as.numeric(readline("Enter attendance percentage: "))
    add_student(name, math_score, science_score, history_score, attendance)
  } else if (choice == 2) {  generate_report()
  } else if (choice == 3) {  cat("Exiting the program.\n")
    break}else
 {cat("Invalid choice. Try again.\n")}}
```

```
6> # Load required library
library(forecast)

# Sales data
sales_data <- data.frame(
  Month = seq(as.Date("2023-01-01"),
as.Date("2023-06-01"), by = "months"),
  Sales = c(12000, 15000, 18000, 16000, 20000,
22000)
)

# Convert to time series (monthly data)
sales_ts <- ts(sales_data$Sales, frequency = 12,
start = c(2023, 1))

# Fit ARIMA model
arima_model <- auto.arima(sales_ts)

# Forecast next 3 months
forecast_result <- forecast(arima_model, h = 3)

# Display forecast and plot
print(forecast_result)
plot(forecast_result)
```