Pseudo-code for login feature

START

FUNCTION login():

DISPLAY "Enter your username: "

INPUT username

DISPLAY "Enter your password: "

INPUT password

// Validate credentials

IF validateCredentials(username, password):

DISPLAY "Login successful!"

RETURN TRUE

ELSE:

DISPLAY "Invalid username or password. Try again."

RETURN FALSE

FUNCTION validateCredentials(username, password):

// Retrieve stored credentials from database or file

STORED\_USERNAME = "exampleUser" // Replace with actual retrieval

STORED\_PASSWORD = "examplePass" // Replace with actual retrieval

IF username == STORED\_USERNAME AND password == STORED\_PASSWORD:

RETURN TRUE

ELSE:

RETURN FALSE

MAIN:

ATTEMPTS = 3

WHILE ATTEMPTS > 0:

SUCCESS = login()

IF SUCCESS:

BREAK

ELSE:

ATTEMPTS = ATTEMPTS - 1

DISPLAY "Remaining attempts: " + ATTEMPTS

IF ATTEMPTS == 0:

DISPLAY "Account locked due to too many failed attempts."

END

Pseudo code for calendar

And adding tasks/ events

START

// Main Menu

FUNCTION mainMenu():

DISPLAY "Calendar Menu:"

DISPLAY "1. View Calendar"

DISPLAY "2. Add Event"

DISPLAY "3. View Events on a Date"

DISPLAY "4. Remove Event"

DISPLAY "5. Exit"

DISPLAY "Choose an option: "

INPUT choice

RETURN choice

// Display the calendar

FUNCTION viewCalendar():

DISPLAY "Displaying calendar for the current month..."

// Logic to generate and display the month's calendar goes here

// Add a new task

FUNCTION addTask(calendar):

DISPLAY "Enter task date (YYYY-MM-DD): "

INPUT taskDate

DISPLAY "Enter task description: "

INPUT taskDescription

IF taskDate NOT IN calendar:

calendar[taskDate] = []

calendar[taskDate].ADD(taskDescription)

DISPLAY "Task added successfully."

// View task on a specific date

FUNCTION viewTasks(calendar):

DISPLAY "Enter date to view tasks(YYYY-MM-DD): "

INPUT taskDate

IF eventDate IN calendar:

DISPLAY "Events on " + eventDate + ":"

FOR event IN calendar[eventDate]:

DISPLAY "- " + event

ELSE:

DISPLAY "No events found for this date."

// Remove an event

FUNCTION removeEvent(calendar):

DISPLAY "Enter date to remove event from (YYYY-MM-DD): "

INPUT eventDate

IF eventDate IN calendar:

DISPLAY "Events on " + eventDate + ":"

FOR i = 1 TO LENGTH(calendar[eventDate]):

DISPLAY i + ". " + calendar[eventDate][i-1]

DISPLAY "Choose an event to remove (number): "

INPUT eventNumber

IF eventNumber >= 1 AND eventNumber <= LENGTH(calendar[eventDate]):

REMOVE calendar[eventDate][eventNumber - 1]

DISPLAY "Event removed successfully."

ELSE:

DISPLAY "Invalid choice."

ELSE:

DISPLAY "No events found for this date."

// Main program logic

FUNCTION main():

DECLARE calendar AS DICTIONARY // e.g., {"YYYY-MM-DD": ["Event 1", "Event 2"]}

WHILE TRUE:

choice = mainMenu()

IF choice == 1:

viewCalendar()

ELSE IF choice == 2:

addEvent(calendar)

ELSE IF choice == 3:

viewEvents(calendar)

ELSE IF choice == 4:

removeEvent(calendar)

ELSE IF choice == 5:

DISPLAY "Exiting calendar. Goodbye!"

BREAK

ELSE:

DISPLAY "Invalid option. Please try again."

END

Pseudo code to send reminders

START

// Initialize a list of tasks with their deadlines

tasks ← [

{ "name": "Task A", "deadline": "2025-01-21 15:00:00" },

{ "name": "Task B", "deadline": "2025-01-21 16:00:00" },

...

]

// Define the time before deadline to send a notification (e.g., 10 minutes)

NOTIFICATION\_THRESHOLD ← 10 MINUTES

// Get the current time

currentTime ← GET\_CURRENT\_TIME()

// Iterate through each task in the list

FOR task IN tasks DO

// Calculate the time difference between the current time and the task's deadline

timeToDeadline ← task["deadline"] - currentTime

// Check if the task is within the notification threshold

IF timeToDeadline <= NOTIFICATION\_THRESHOLD AND timeToDeadline > 0 THEN

// Send a notification

SEND\_NOTIFICATION("Reminder: " + task["name"] + " is about to finish!")

ENDIF

END FOR

END