

Address Book Shell Script — Full Guide (Code + Explanation + Theory + Viva)

Aim

To implement a menu-driven Bash shell script that creates, views, adds, deletes, and modifies records in a text-based address book.

Software/Tools

Linux/Ubuntu with Bash shell, basic GNU utilities (grep, sed, cat, touch).

Complete Script

```
#!/bin/bash

FILE="addressbook.txt"

# 1) Create Address Book
createBook() {
    if [ -e "$FILE" ]; then
        echo "Address book already exists."
    else
        touch "$FILE"
        echo "Address book created."
    fi
}

# 2) View Address Book
viewBook() {
    if [ ! -s "$FILE" ]; then
        echo "Address book is empty."
    else
        echo "----- Address Book -----"
        cat "$FILE"
    fi
}

# 3) Add a Record
addRecord() {
    echo "Enter: LastName,FirstName,Email,Phone"
    read record
    echo "$record" >> "$FILE"
    echo "Record added."
}

# 4) Delete a Record
deleteRecord() {
    echo "Enter name or email to search:"
    read search
    grep -n "$search" "$FILE"          # show matching lines with numbers

    echo "Enter line number to delete:"
    read line
    sed -i "${line}d" "$FILE"          # delete that line
    echo "Record deleted."
}

# 5) Modify a Record
modifyRecord() {
```

```

    echo "Enter name or email to search:"
    read search
    grep -n "$search" "$FILE"

    echo "Enter line number to change:"
    read line
    echo "Enter new record (LastName,FirstName,Email,Phone):"
    read new

    sed -i "${line}s/./$new/" "$FILE"      # replace full line
    echo "Record updated."
}

# 6) Exit
exitProgram() {
    echo "Goodbye!"
    exit 0
}

# Menu Loop
while true; do
    echo "----- Address Book Menu -----"
    echo "1. Create Address Book"
    echo "2. View Address Book"
    echo "3. Add Record"
    echo "4. Delete Record"
    echo "5. Modify Record"
    echo "6. Exit"
    echo "Enter your choice: "
    read choice

    case $choice in
        1) createBook ;;
        2) viewBook ;;
        3) addRecord ;;
        4) deleteRecord ;;
        5) modifyRecord ;;
        6) exitProgram ;;
        *) echo "Invalid choice! Please pick 1-6." ;;
    esac
done

```

Explanation (Key Points)

- Shebang (`#!/bin/bash`) ensures the script runs with the Bash interpreter.
- `FILE` variable stores the data file (`addressbook.txt`) used as a simple database.
- `createBook()`: uses `-e` to test existence; `touch` creates the file if missing.
- `viewBook()`: uses `-s` to check if file has content; `cat` prints the records.
- `addRecord()`: reads a CSV-style line and appends with `>>` to the file.
- `deleteRecord()`: `grep -n` shows matches with line numbers; `sed -i "Nd"` deletes that line.
- `modifyRecord()`: `sed -i "Ns./NEW/"` replaces the entire Nth line with new data.
- `exitProgram()`: clean exit with status code 0.
- The menu uses `while true` loop and `case...esac` to dispatch functions.

How to Run

- 1) Save the file as addressbook.sh
- 2) Open terminal in the same folder
- 3) Make executable (optional): `chmod +x addressbook.sh`
- 4) Run with: `bash addressbook.sh` (or `./addressbook.sh` after `chmod +x`)

Theory for Practical Exam

- 1 A shell script is a text program executed by a Unix shell to automate tasks.
- 2 This experiment demonstrates a menu-driven program using functions, loops, conditionals, and file handling.
- 3 Data is stored in a flat text file in CSV-like format: LastName,FirstName,Email,Phone.
- 4 Key commands: touch (create file), cat (display), grep -n (search with line numbers), sed -i (in-place edit), read (user input).
- 5 Testing expressions: -e checks existence, -s checks non-empty file.
- 6 Advantages: quick to implement, portable across Linux distros; Limitations: no schema validation, concurrent edits not handled.

Viva Questions & Answers

- Q: What does the shebang do?
Ans: It specifies the interpreter (here, /bin/bash) to execute the script.
- Q: Difference between > and >> ?
Ans: > overwrites a file; >> appends to the end of the file.
- Q: Meaning of -e and -s in test?
Ans: -e: file exists; -s: file exists and size > 0.
- Q: How do you delete the Nth line with sed?
Ans: sed -i "\${N}d" filename
- Q: How do you replace the Nth line completely?
Ans: sed -i "\${N}s/.*/NEW/" filename
- Q: Why use grep -n before delete/modify?
Ans: To display matching lines with numbers so the user can choose the correct record safely.
- Q: What is case...esac?
Ans: A multi-branch selection similar to switch-case in other languages.
- Q: How to make a script executable?
Ans: chmod +x script.sh and then run with ./script.sh
- Q: How to handle commas in input safely?
Ans: Quote variables ("record") and ensure consistent CSV input.
- Q: One limitation of this approach?
Ans: No validation or concurrency control; errors may occur if two users edit simultaneously.

Result

The script successfully implements CRUD operations on a text-based address book using Bash.