

## **Objective:**

To develop a shell script that maintains and analyzes student attendance records efficiently using file operations, loops, and conditional logic in Bash.

### **1. Problem Explanation**

Maintaining attendance manually is time-consuming and prone to human errors.

This project automates attendance management using **UNIX Shell Scripting**, enabling teachers to:

- Mark daily attendance efficiently
- View student records anytime
- Generate attendance summaries and percentages
- Identify defaulters automatically
- Receive alerts for consecutive absences

The project demonstrates how basic shell commands like `echo`, `grep`, `awk`, and `case` can be combined to build a **fully functional CLI-based attendance tracking system**.

### **2. TOOLS & ENVIRONMENT**

Tools	Purpose
Git Bash / Linux Terminal	To execute Bash commands
Text Editor (VS Code / Nano)	For writing and editing the script
Bash Shell ( $\geq 4.0$ )	Script execution environment
Files Used	<code>students.txt</code> , <code>attendance.txt</code>

### **3. ALGORITHM / FLOW**

#### **Step-by-Step Logic:**

##### **1. Initialize Files**

- Check if `students.txt` exists; if not, create it with default students.
- Create or verify `attendance.txt` for storing daily attendance.

##### **2. Display Menu**

- Show available options (1 to 10).

##### **3. Mark Attendance**

- Input date and status (P/A) for each student.

Save record in the format:

`YYYY-MM-DD, StudentID, Name, Status`

##### **4. View Section Info**

- Display all student IDs and names in table form.
- Show total number of students.

##### **5. View Full Attendance Log**

- Display complete history from `attendance.txt`.

##### **6. Search Student**

- Allow search by **ID** or **Name**.
- Show detailed report with total, present, absent, and percentage.

##### **7. Attendance Summary**

For each student, show:

`ID | Name | Present | Absent | Percentage`

##### **8. Attendance Percentage (Class)**

- Show attendance percentage for every student in summary format.

#### 9. Defaulter List

- Display students whose attendance percentage < threshold (default 40%).

#### 10. Alert System

- Detect and show warnings for students with **3 consecutive absences**.

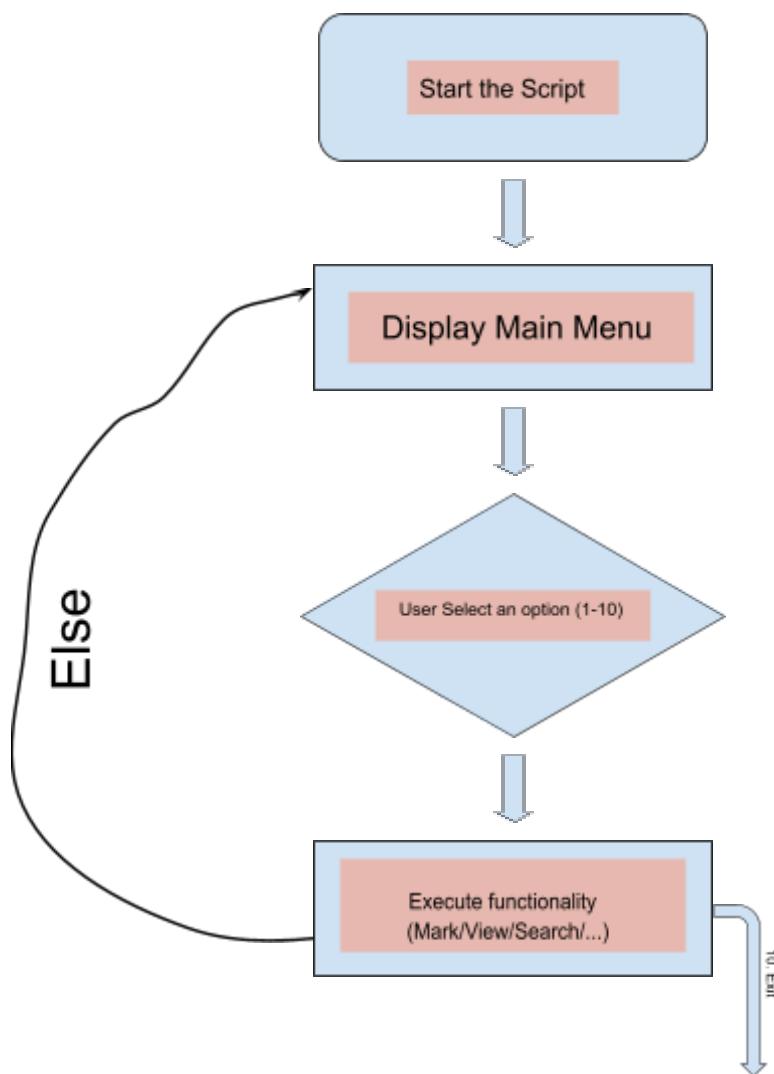
#### 11. Add / Remove Student

- Add new students with ID and Name.
- Remove students either by ID or Name.

#### 12. Exit

- End the script gracefully.

### 4. FLOWCHART



## 5. Code

```
attendance_file="attendance.txt"
students_file="students.txt"
threshold=40

# Default students if not exists
if [ ! -f "$students_file" ]; then
    echo -e "S101,Azman\nS102,Nusrat\nS103,Rahim\nS104,Sadia\nS105,Arif" >
"$students_file"
fi

load_students() {
    students=()
    while IFS= read -r line; do
        students+=("$line")
    done < "$students_file"
}

show_menu() {
    echo -----
    echo " STUDENT ATTENDANCE TRACKER "
    echo "1. Mark Attendance"
    echo "2. View Section Info"
    echo "3. View Full Attendance Log"
    echo "4. Search Student (Full Report)"
    echo "5. Attendance Summary"
    echo "6. Attendance Percentage"
    echo "7. Defaulter List"
    echo "8. Alert System (3 consecutive absences)"
    echo "9. Add/Remove Student"
    echo "10. Exit"
    echo -----
}

while true; do
    load_students
    show_menu
    read -p "Enter your choice: " choice

    case $choice in
        1)
            read -p "Enter Date (YYYY-MM-DD): " date
            echo "Mark attendance: P for present, A for absent"
            for student in "${students[@]}"; do
                IFS=',' read -r sid sname <<< "$student"
                while true; do

```

```

read -p "Status for $sname ($sid) (P/A): " status
status=$(echo "$status" | tr 'a-z' 'A-Z')
if [[ "$status" == "P" || "$status" == "A" ]]; then
    echo "$date,$sid,$sname,$status" >> "$attendance_file"
    break
else
    echo "Invalid input! Please enter P or A."
fi
done
done
echo "✅ Attendance saved."
;;

```

2)

```

echo "---- Section Students List ----"
echo "ID      | Name"
echo "-----+-----"
awk -F, '{printf "%-7s | %s\n", $1, $2}' "$students_file"
total_students=$(wc -l < "$students_file")
echo "-----"
echo "Total Students: $total_students"
;;

```

3)

```

echo "---- Full Attendance Log ----"
if [ ! -f "$attendance_file" ]; then
    echo "No attendance records found."
else
    cat "$attendance_file"
fi
;;

```

4)

```

echo "Search by: 1) ID 2) Name"
read -p "Choose 1 or 2: " search_choice
if [[ "$search_choice" == "1" ]]; then
    read -p "Enter Student ID: " sid_search
    student_info=$(grep -i "^${sid_search}" "$students_file")
elif [[ "$search_choice" == "2" ]]; then
    read -p "Enter Student Name: " name_search
    student_info=$(grep -i ".*$${name_search}.*" "$students_file" | head -n1)
fi

if [ -z "$student_info" ]; then
    echo "Student not found!"
    continue
fi

```

```

target_id=$(echo "$student_info" | cut -d, -f1)
target_name=$(echo "$student_info" | cut -d, -f2)

echo "=====
echo " A-to-Z Report for: $target_name ($target_id)"
echo "====="

awk -F, -v id="$target_id" '
{
    date=$1; sid=$2; status=$4
    if (sid == id) {
        total++; if (status == "P") present++
        split(date, d, "-"); month=d[1]"-"d[2]
        month_total[month]++; if (status=="P") month_present[month]++
    }
}
END {
    perc=(total>0)?(present/total)*100:0
    printf "Summary: %d Present / %d Total Days\n", present, total
    printf "Percentage: %.2f%%\n", perc
    print "\n--- Monthly Breakdown ---"
    n=asorti(month_total, sm)
    for(i=1;i<=n;i++){
        m=sm[i]; abs=month_total[m]-month_present[m]
        printf "%-10s: %d Present, %d Absent (%d Total)\n", m, month_present[m], abs,
month_total[m]
    }
} '$attendance_file'
;;

```

5)

```

echo "---- Attendance Summary ----"
awk -F, '
BEGIN{while((getline<"$students_file">0){split($0,s,"");id=s[1];name=s[2];t[id]=0;p[id]=0;n[i
d]=name})
{sid=$2;st=$4;if(sid in n){t[sid]++;if(st=="P")p[sid]++}}
END{for(i in n){printf "%s (%s): %d/%d days\n", n[i], i, p[i], t[i]}}
' "$attendance_file"
;;

```

6)

```

echo "---- Attendance Percentage ----"
awk -F, '
BEGIN{while((getline<"$students_file">0){split($0,s,"");id=s[1];name=s[2];t[id]=0;p[id]=0;n[i
d]=name})
{sid=$2;st=$4;if(sid in n){t[sid]++;if(st=="P")p[sid]++}}

```

```

END{for(i in n){perc=(t[i]>0)?(p[i]/t[i])*100:0;printf "%s (%s): %.2f%%\n",n[i],i,perc}}
' "$attendance_file"
;;
7)
echo "---- Defaulter List (< $threshold%) ----"
awk -F, -v t=$threshold '
BEGIN{while((getline<"$students_file">0){split($0,s,"");id=s[1];name=s[2];t[id]=0;p[id]=0;n[i]
d]=name}}
{sid=$2;st=$4;if(sid in n){t[sid]++;if(st=="P")p[sid]++}}
END{for(i in n){perc=(t[i]>0)?(p[i]/t[i])*100:0;if(perc<t)printf "%s (%s): %.2f%%\n",n[i],i,perc}}
' "$attendance_file"
;;
8)
echo "---- Alert System (3 consecutive absences) ----"
sort -t, -k1,1 "$attendance_file" | awk -F,
{sid=$2;name=$3;st=$4;if(sid!=last){c=0;if(st=="A"){c++}else{c=0}
if(c==3){print "⚠ ALERT: " name " (" sid ") - 3 consecutive absences!"}
last=sid}
;;
9)
echo "---- Add/Remove Student ----"
echo "a) Add Student"
echo "b) Remove Student"
read -p "Choose action (a/b): " act
if [[ "$act" == "a" ]]; then
    read -p "Enter Student ID: " nid
    read -p "Enter Student Name: " nname
    echo "$nid,$nname" >> "$students_file"
    echo "Added $nname ($nid)"
elif [[ "$act" == "b" ]]; then
    read -p "Enter Student ID to remove: " rid
    grep -v -i "^$rid," "$students_file" > tmp && mv tmp "$students_file"
    echo "Removed ID: $rid"
fi
;;
10) echo "Exiting..."; exit 0;;
*) echo "Invalid choice!";;
esac
echo
done

```

## 6. SAMPLE INPUT/OUTPUT

### Sample 1 — Marking Attendance

```
Enter Date (YYYY-MM-DD): 2025-11-07
Mark attendance: P for present, A for absent
Status for Azman (S101): P
Status for Nusrat (S102): A
...
Attendance saved.
```

### Sample 2 — Viewing Section Info

```
---- Section Students List ----
ID      | Name
-----+-----
S101   | Azman
S102   | Nusrat
S103   | Rahim
S104   | Sadia
S105   | Arif
-----
Total Students: 5
```

### Sample 3 — Attendance Summary

```
----- Attendance Summary -----
ID      | Name      | P | A | %
-----+
S101   | Azman     | 7 | 1 | 87.50%
S102   | Nusrat     | 5 | 3 | 62.50%
S103   | Rahim     | 3 | 5 | 37.50%
-----+
```

### Sample 4 — Defaulter List (<40%)

```
----- Defaulter List (<40%) -----
S103 | Rahim | 37% (3/8)
```

### Sample 5 — Alert System

```
---- Alert: Students with 3 consecutive absences ----
S103 | Rahim has 3 consecutive absences (last on 2025-11-06)
```

## 7. OUTPUT FILE STRUCTURE

### students.txt

S101 , Azman

S102 , Nusrat

S103 , Rahim

S104 , Sadia

S105 , Arif

### attendance.txt

2025-11-07 , S101 , Azman , P

2025-11-07 , S102 , Nusrat , A

2025-11-07 , S103 , Rahim , P

2025-11-08 , S101 , Azman , P

2025-11-08 , S102 , Nusrat , A

## 8. CONCLUSION

Successfully implemented a **menu-driven attendance tracker** in Bash that:

- Handles dynamic student data.
- Tracks daily attendance.
- Calculates percentages and identifies defaulters.
- Provides alerts for consecutive absences.
- Demonstrates efficient use of **file handling, conditionals, loops, and string operations** in UNIX shell scripting.