

Experiment 12

Experiment 12: Building a Rule-Based Expert System

Name: Hrithvik Bhardwaj

Roll No.: 590029169

Date: 2025-11-30

Aim

To develop a rule-based expert system using shell scripting, and automate system tasks using cron scheduling and system administration scripts.

Requirements

- Linux system with Bash shell
 - Text editor (nano/vim)
 - Access to cron
 - Basic shell scripting knowledge
-

Theory

1. Process Automation & Job Scheduling

Linux provides tools to automate tasks:

cron (Recurring tasks)

```
minute hour day month day_of_week command
```

Example:

```
0 9 * * * /home/user/backup.sh
```

at (One-time tasks)

```
echo "/home/user/script.sh" | at 2:00 AM
```

2. System Administration Scripts

Used for automated:

- Backups

- User management
 - Log monitoring
 - Health checks
-

3. Services and Daemons

Managed using systemctl:

```
systemctl start service
systemctl stop service
systemctl enable service
systemctl status service
```

Procedure & Observations

Students created a rule-based medical expert system using:

- If-else logic
 - Pattern matching
 - Multi-symptom handling
 - Cron scheduling
-

LAB EXERCISE

Exercise: Basic Medical Expert System

Task Statement

Create a simple expert system that gives recommendations based on user symptoms.

Commands

```
#!/bin/bash
echo "Welcome to the Medical Expert System"
echo "Enter your symptoms:"
read symptoms

if [[ "$symptoms" == *"fever"* ]]; then
    echo "Recommendation: Take fever reducer and rest."
fi
if [[ "$symptoms" == *"cough"* ]]; then
    echo "Recommendation: Drink warm fluids."
fi
if [[ "$symptoms" == *"cold"* ]]; then
```

```

echo "Recommendation: Use decongestants."
fi

if [[ "$symptoms" != *"fever"* && "$symptoms" != *"cough"* && "$symptoms" != *"cold"* ]]; then
    echo "Consult a doctor."
fi

```

Output

```

retr0@Retr0:~$ cat expert.sh
#!/bin/bash
echo "Welcome to the Medical Expert System"
echo "Enter your symptoms:"
read symptoms

if [[ "$symptoms" == *"fever"* ]]; then
    echo "Recommendation: Take fever reducer and rest."
fi
if [[ "$symptoms" == *"cough"* ]]; then
    echo "Recommendation: Drink warm fluids."
fi
if [[ "$symptoms" == *"cold"* ]]; then
    echo "Recommendation: Use decongestants."
fi
if [[ "$symptoms" != *"fever"* && "$symptoms" != *"cough"* && "$symptoms" != *"cold"* ]]; then
    echo "Consult a doctor."
fi

retr0@Retr0:~$ chmod +x expert.sh
retr0@Retr0:~$ ./expert.sh
Welcome to the Medical Expert System
Enter your symptoms: fever cough
Recommendation: Take fever reducer and rest.
Recommendation: Drink warm fluids.

```

ASSIGNMENTS

Assignment 1: Extended Expert System

Task Statement

Improve the expert system with multiple symptoms and detailed recommendations.

Commands

```

#!/bin/bash
echo "Enter symptoms separated by commas:"
read symptoms

symptoms_lower=$(echo "$symptoms" | tr '[:upper:]' '[:lower:]')

if [[ "$symptoms_lower" == *"fever"* ]]; then
    echo "• Take medication"
    echo "• Monitor temperature"
fi
if [[ "$symptoms_lower" == *"fatigue"* ]]; then
    echo "• Get proper sleep"
fi
if [[ "$symptoms_lower" == *"nausea"* ]]; then
    echo "• Drink ginger tea"
fi

```

Output

```

retr0@Retr0:~$ cat extended.sh
#!/bin/bash
echo "Enter symptoms separated by commas:"
read symptoms

symptoms_lower=$(echo "$symptoms" | tr '[:upper:]' '[:lower:]')

if [[ "$symptoms_lower" == *"fever"* ]]; then
    echo "• Take medication"
    echo "• Monitor temperature"
fi
if [[ "$symptoms_lower" == *"fatigue"* ]]; then
    echo "• Get proper sleep"
fi
if [[ "$symptoms_lower" == *"nausea"* ]]; then
    echo "• Drink ginger tea"
fi

retr0@Retr0:~$ chmod +x extended.sh
retr0@Retr0:~$ ./extended.sh
Enter symptoms separated by commas: fever,nausea
• Take medication
• Monitor temperature
• Drink ginger tea

```

Assignment 2: Scheduling Using Cron

Task Statement

Schedule the expert system to run daily at 8 AM.

Commands

```
crontab -e
```

Add:

```
0 8 * * * /path/to/expert_system.sh
```

Make executable:

```
chmod +x /path/to/expert_system.sh
```

Output

```
retr0@Retr0:~$ crontab -e
0 8 * * * /home/retr0/expert.sh
retr0@Retr0:~$ crontab -l
0 8 * * * /home/retr0/expert.sh
```

Assignment 3: Multi-Recommendation System

Task Statement

Use associative arrays to provide structured recommendations.

Commands

```
#!/bin/bash
declare -A recommendations
recommendations["fever"]="Take medication|Stay hydrated"
recommendations["cold"]="Rest|Use nasal spray"

symptoms=$(echo "$input" | tr ',' ' ')
for s in $symptoms; do
    echo "For $s:"
    IFS='|' read -ra tips <<< "${recommendations[$s]}"
    for t in "${tips[@]}"; do
        echo "- $t"
    done
done
```

Output

```
retr0@Retr0:~$ cat multi.sh
#!/bin/bash
declare -A recommendations
recommendations["fever"]="Take medication|Stay hydrated"
recommendations["cold"]="Rest|Use nasal spray"

echo "Enter symptoms separated by commas:"
read input

symptoms=$(echo "$input" | tr ',' ' ')
for s in $symptoms; do
    echo "For $s:"
    IFS='|' read -ra tips <<< "${recommendations[$s]}"
    for t in "${tips[@]}"; do
        echo "- $t"
    done
done

retr0@Retr0:~$ chmod +x multi.sh
retr0@Retr0:~$ ./multi.sh
Enter symptoms separated by commas: fever,cold
For fever:
- Take medication
- Stay hydrated
For cold:
- Rest
- Use nasal spray
```

Result

The rule-based expert system was successfully built.
Cron scheduling and automation concepts were demonstrated.

Conclusion

This experiment taught rule-based logic, automation, and expert system development using shell scripting.