Experiment 11: Shell Programming – Advanced String & File Operations

Name: Aditya Mishra

Roll No.: 590029219

Date: 2025-11-05

Aim

To write and execute shell scripts for advanced string manipulation, file processing, menu-driven systems, and dictionary-based word validation.

Requirements

- Linux system with Bash shell
- Commands: `grep`, `cut`, `awk`, `rev`, `tr`, `df`, `free`, `cal`, `date`
- Dictionary file at `/usr/share/dict/words` (for dictionary check task)

Exercise 1: Split Sentence into Words

Task Statement

Write a shell script to split a sentence into individual words and print them one per line.

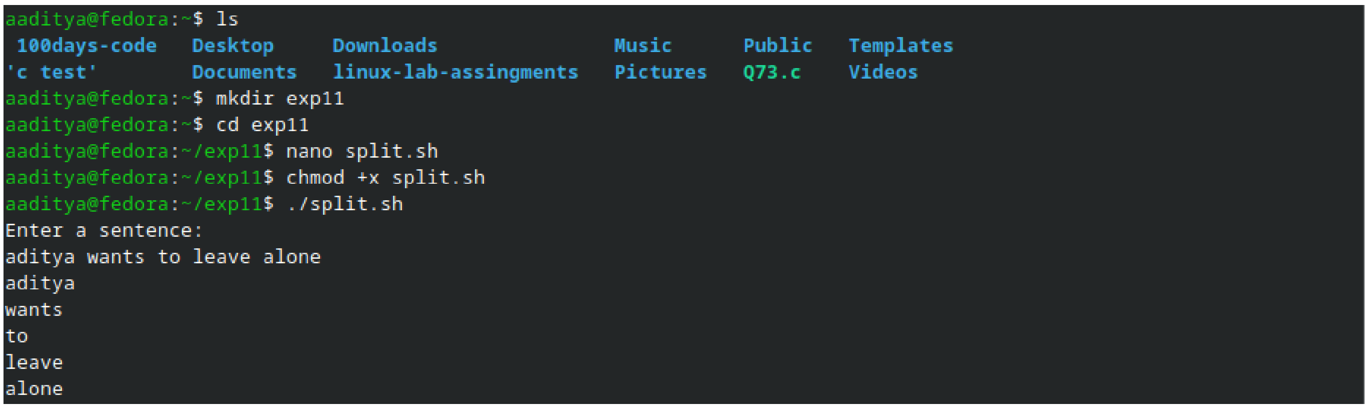
Script

```
#!/bin/bash
echo "Enter a sentence:"
read sentence
for word in $sentence; do
echo "$word"
done
```

Explanation

- When `$sentence` is **unquoted**, Bash performs word splitting automatically based on the **IFS (Internal Field Separator)**.
- By default, it splits on spaces, tabs, and newlines.
- Each word becomes a separate iteration of the loop.

Output



```
aaditya@fedora:~$ ls
 100days-code   Desktop     Downloads               Music      Public   Templates
'c test'        Documents   linux-lab-assingments   Pictures   Q73.c    Videos
aaditya@fedora:~$ mkdir exp11
aaditya@fedora:~$ cd exp11
aaditya@fedora:~/exp11$ nano split.sh
aaditya@fedora:~/exp11$ chmod +x split.sh
aaditya@fedora:~/exp11$ ./split.sh
Enter a sentence:
aditya wants to leave alone
aditya
wants
to
leave
alone
```

Exercise 2: Palindrome Check

Task Statement

Write a script that checks whether a given string is a palindrome.

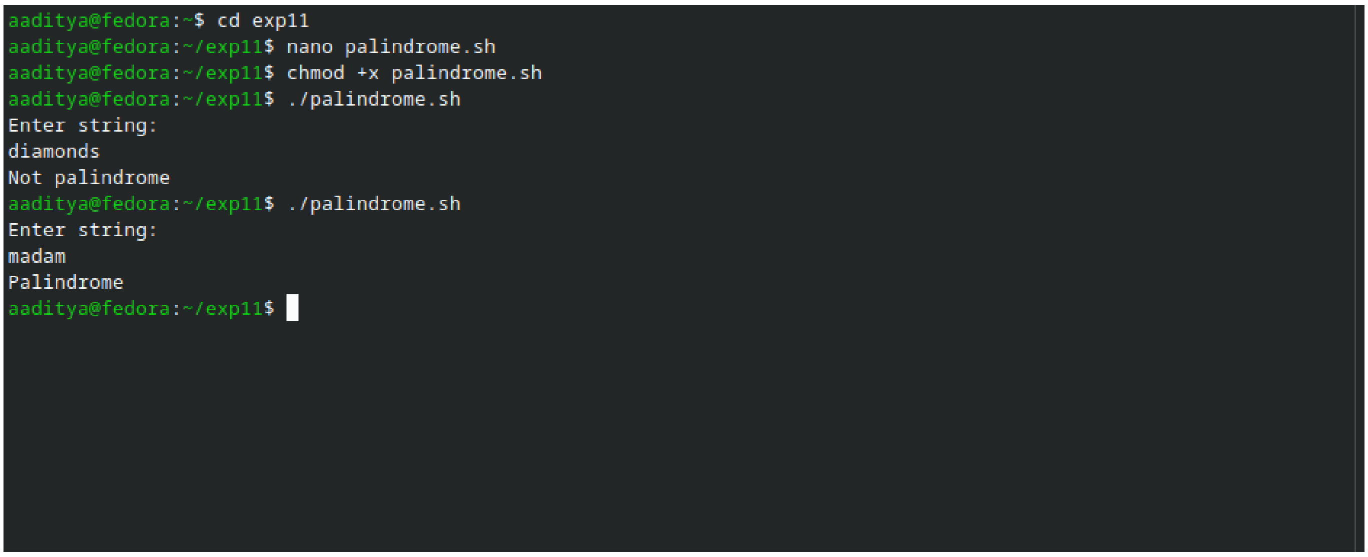
Script

```
#!/bin/bash
echo "Enter string:"
read str
rev=$(echo "$str" | rev)
if [ "$str" = "$rev" ]; then
echo "Palindrome"
else
echo "Not palindrome"
fi
```

Explanation

- The `rev` command reverses a string character by character.
- Comparison checks if the original string equals the reversed one.
- This comparison is **case-sensitive** and includes spaces.

Output



```
aaditya@fedora:~$ cd exp11
aaditya@fedora:~/exp11$ nano palindrome.sh
aaditya@fedora:~/exp11$ chmod +x palindrome.sh
aaditya@fedora:~/exp11$ ./palindrome.sh
Enter string:
diamonds
Not palindrome
aaditya@fedora:~/exp11$ ./palindrome.sh
Enter string:
madam
Palindrome
aaditya@fedora:~/exp11$
```

Exercise 3: CSV File Processing – Print First Column

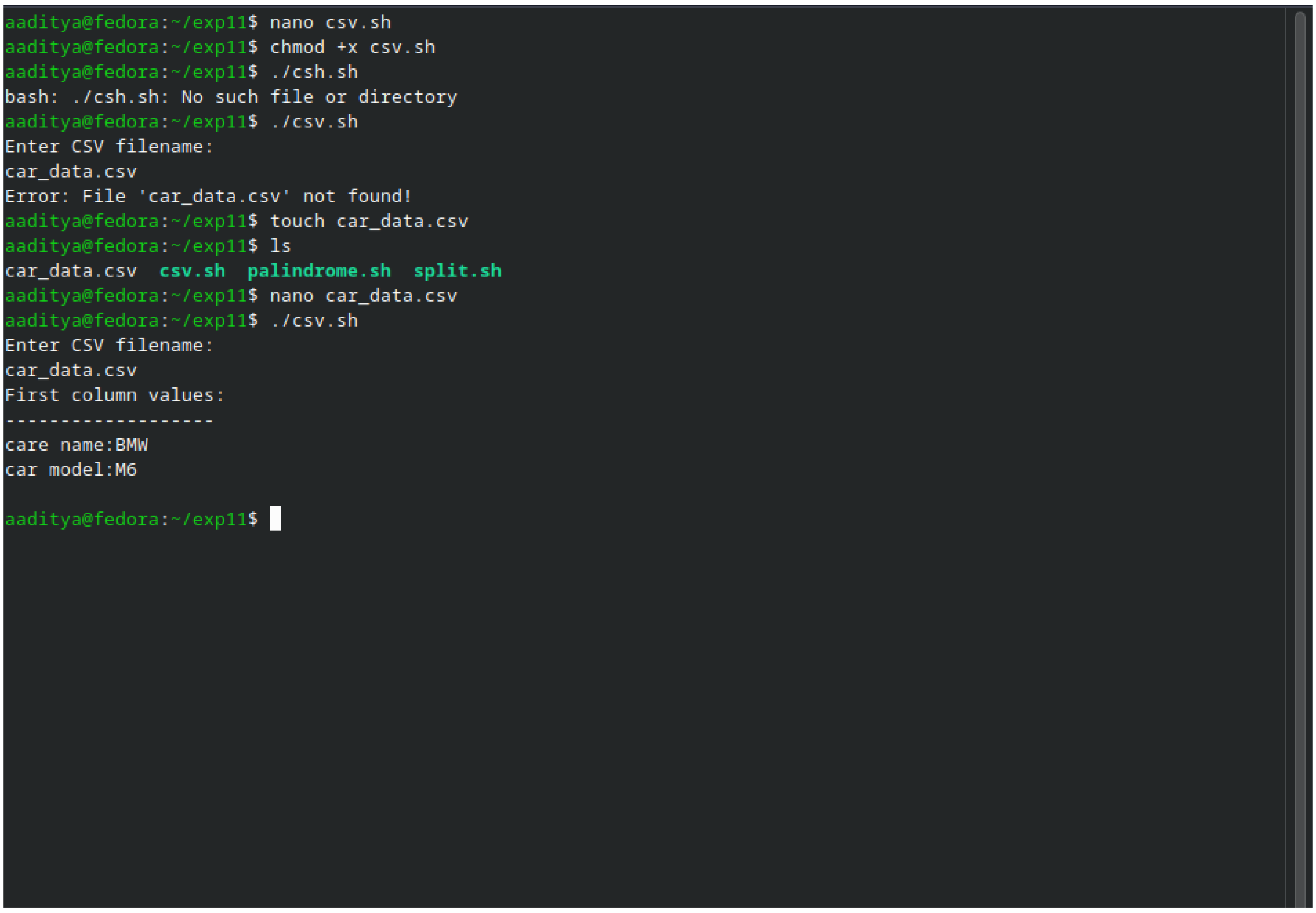
Task Statement

Write a script to print the first column of a CSV file provided by the user.

Script

```
#!/bin/bash
echo "Enter CSV filename:"
read filename
if [ ! -f "$filename" ]; then
echo "Error: File '$filename' not found!"
exit 1
fi
echo "First column values:"
echo "-------------------"
cut -d',' -f1 "$filename"
```

Output



```
aaditya@fedora:~/exp11$ nano csv.sh
aaditya@fedora:~/exp11$ chmod +x csv.sh
aaditya@fedora:~/exp11$ ./csh.sh
bash: ./csh.sh: No such file or directory
aaditya@fedora:~/exp11$ ./csv.sh
Enter CSV filename:
car_data.csv
Error: File 'car_data.csv' not found!
aaditya@fedora:~/exp11$ touch car_data.csv
aaditya@fedora:~/exp11$ ls
car_data.csv  csv.sh  palindrome.sh  split.sh
aaditya@fedora:~/exp11$ nano car_data.csv
aaditya@fedora:~/exp11$ ./csv.sh
Enter CSV filename:
car_data.csv
First column values:
-------------------
care name:BMW
car model:M6

aaditya@fedora:~/exp11$ 
```

Exercise 4: Interactive Menu System

Task Statement

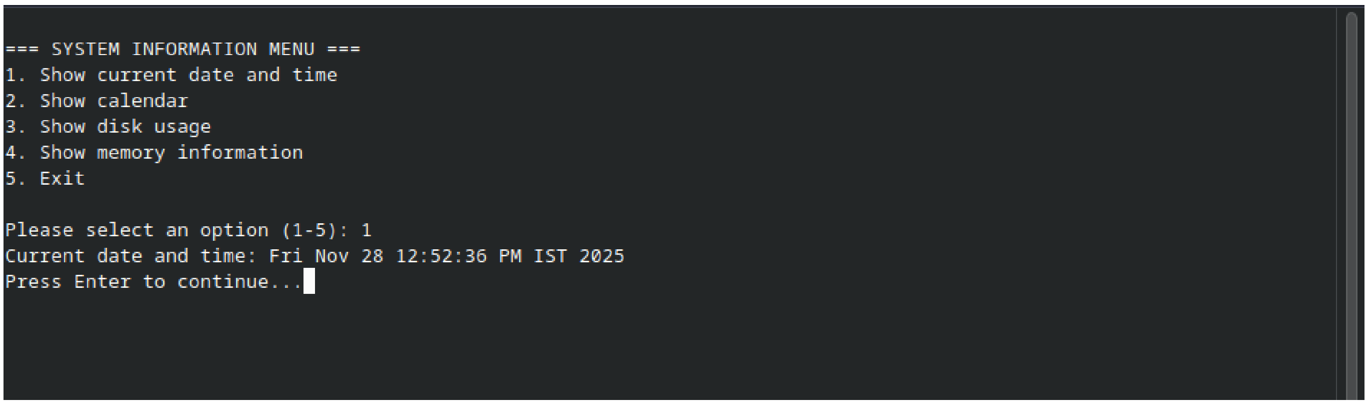
Create a menu-driven script that allows users to display system information.

Script

```
#!/bin/bash
show_date() {
echo "Current date and time: $(date)"
}
show_calendar() {
echo "Current month calendar:"
cal
}
show_disk_usage() {
echo "Disk usage:"
df -h
}
show_memory_info() {
echo "Memory information:"
free -h
}
```

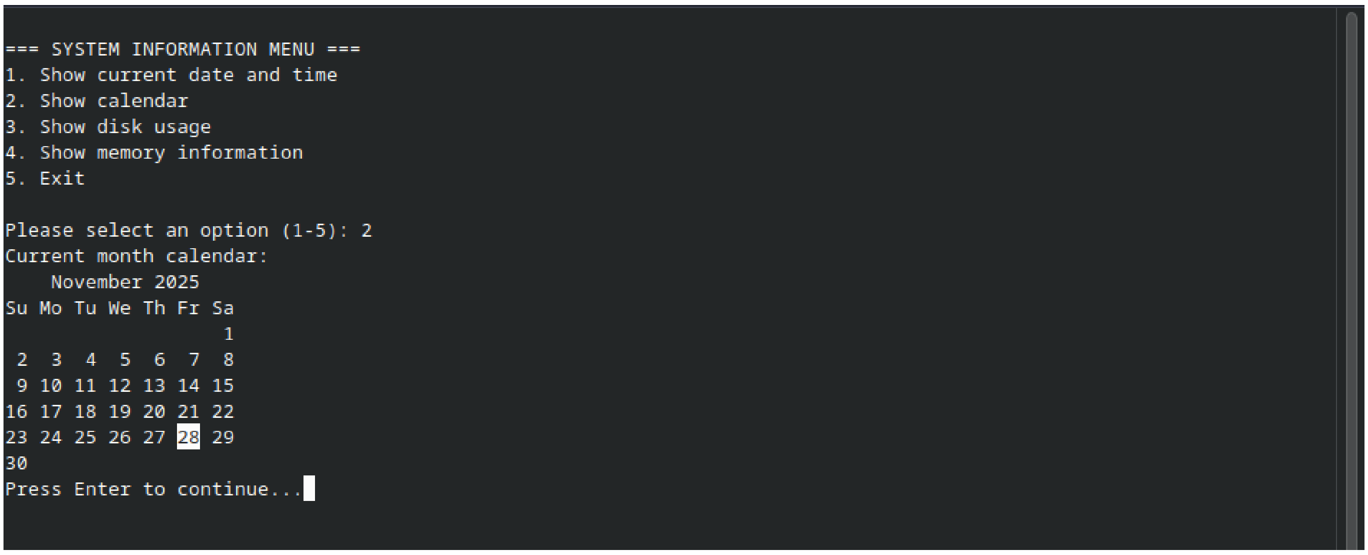
```
while true; do
echo ""
echo "=== SYSTEM INFORMATION MENU ==="
echo "1. Show current date and time"
echo "2. Show calendar"
echo "3. Show disk usage"
echo "4. Show memory information"
echo "5. Exit"
echo ""
read -p "Please select an option (1-5): " choice
case $choice in
1) show_date ;;
2) show_calendar ;;
3) show_disk_usage ;;
4) show_memory_info ;;
5) echo "Goodbye!"; break ;;
*) echo "Invalid option! Please enter a number between 1-5." ;;
esac
read -p "Press Enter to continue..."
clear
done
```

Output



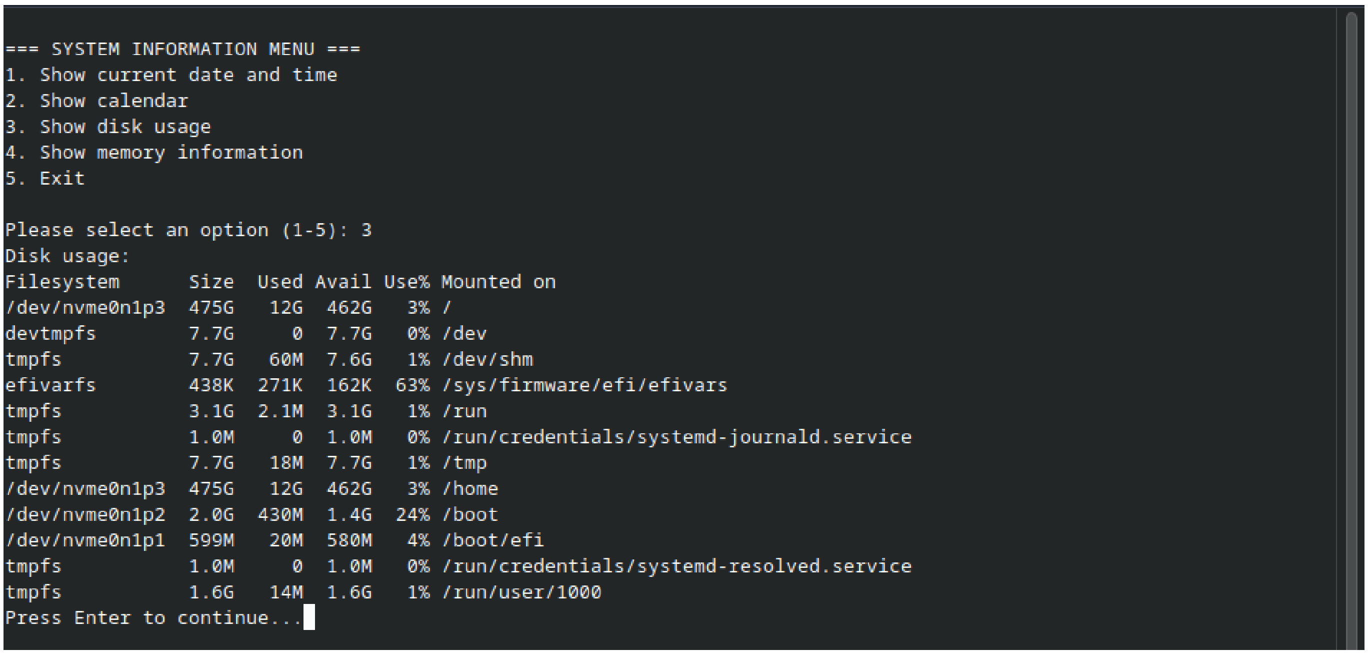
```
=== SYSTEM INFORMATION MENU ===
1. Show current date and time
2. Show calendar
3. Show disk usage
4. Show memory information
5. Exit

Please select an option (1-5): 1
Current date and time: Fri Nov 28 12:52:36 PM IST 2025
Press Enter to continue...
```



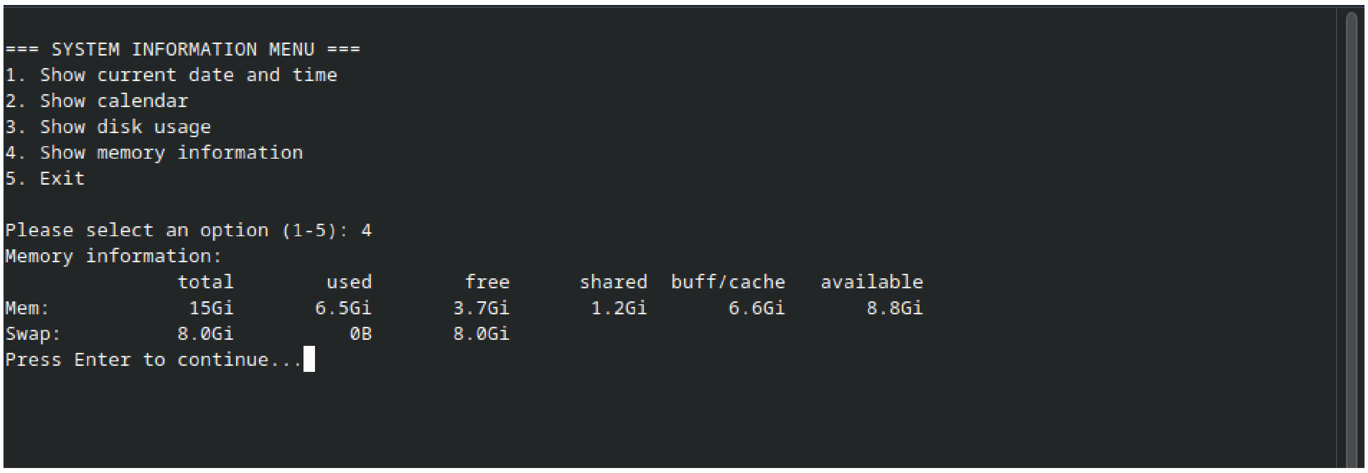
```
=== SYSTEM INFORMATION MENU ===
1. Show current date and time
2. Show calendar
3. Show disk usage
4. Show memory information
5. Exit

Please select an option (1-5): 2
Current month calendar:
    November 2025
Su Mo Tu We Th Fr Sa
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
Press Enter to continue...
```



```
=== SYSTEM INFORMATION MENU ===
1. Show current date and time
2. Show calendar
3. Show disk usage
4. Show memory information
5. Exit

Please select an option (1-5): 3
Disk usage:
Filesystem      Size  Used Avail Use% Mounted on
/dev/nvme0n1p3  475G   12G  462G   3% /
devtmpfs        7.7G     0  7.7G   0% /dev
tmpfs           7.7G   60M  7.6G   1% /dev/shm
efivarfs        438K  271K  162K  63% /sys/firmware/efi/efivars
tmpfs           3.1G  2.1M  3.1G   1% /run
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-journald.service
tmpfs           7.7G   18M  7.7G   1% /tmp
/dev/nvme0n1p3  475G   12G  462G   3% /home
/dev/nvme0n1p2  2.0G  430M  1.4G  24% /boot
/dev/nvme0n1p1  599M   20M  580M   4% /boot/efi
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-resolved.service
tmpfs           1.6G   14M  1.6G   1% /run/user/1000
Press Enter to continue...
```



```
=== SYSTEM INFORMATION MENU ===
1. Show current date and time
2. Show calendar
3. Show disk usage
4. Show memory information
5. Exit

Please select an option (1-5): 4
Memory information:
               total        used        free      shared  buff/cache   available
Mem:            15Gi       6.5Gi       3.7Gi       1.2Gi       6.6Gi       8.8Gi
Swap:          8.0Gi          0B       8.0Gi
Press Enter to continue...
```

Exercise 5: Dictionary Word Check

Task Statement

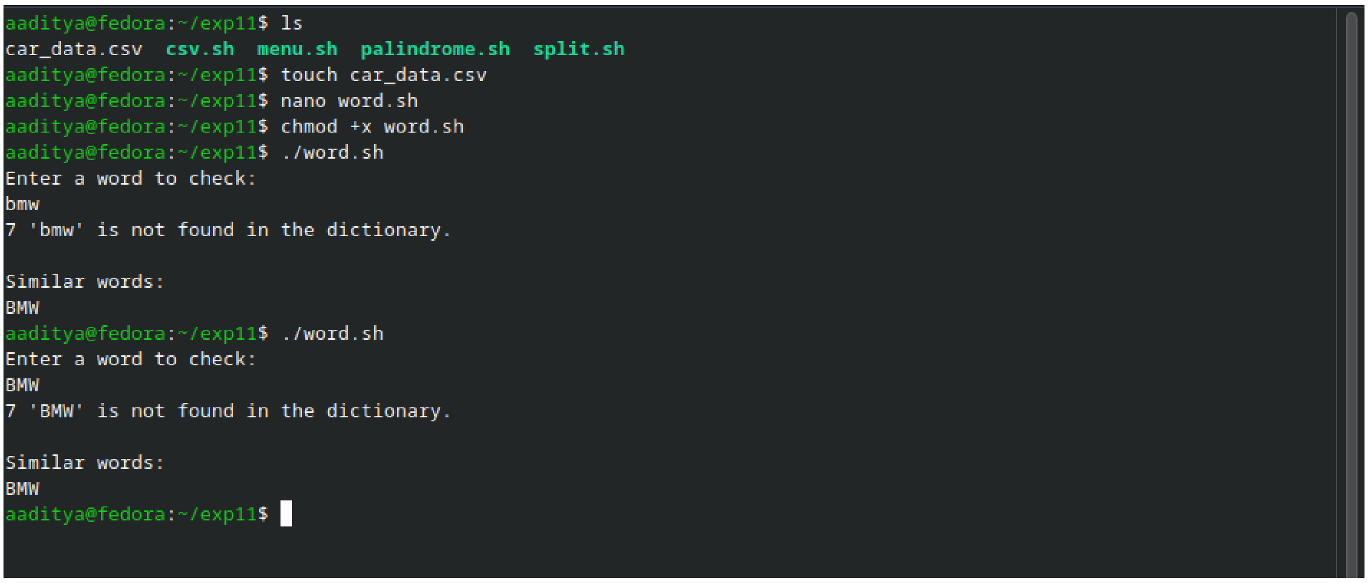
Write a script to check if a word exists in the system dictionary.

Script

```
#!/bin/bash
DICTIONARY="/usr/share/dict/words"
if [ ! -f "$DICTIONARY" ]; then
echo "Error: Dictionary file not found!"
exit 1
fi
echo "Enter a word to check:"
read word
word_lower=$(echo "$word" | tr '[:upper:]' '[:lower:]')
if grep -q "^${word_lower}$" "$DICTIONARY"; then
echo "3 '$word' is a valid English word."
```

```
else
echo "7 '$word' is not found in the dictionary."
echo ""
echo "Similar words:"
grep -i "^${word:0:3}" "$DICTIONARY" | head -5
fi
```

output:



```
aaditya@fedora:~/exp11$ ls
car_data.csv  csv.sh  menu.sh  palindrome.sh  split.sh
aaditya@fedora:~/exp11$ touch car_data.csv
aaditya@fedora:~/exp11$ nano word.sh
aaditya@fedora:~/exp11$ chmod +x word.sh
aaditya@fedora:~/exp11$ ./word.sh
Enter a word to check:
bmw
7 'bmw' is not found in the dictionary.

Similar words:
BMW
aaditya@fedora:~/exp11$ ./word.sh
Enter a word to check:
BMW
7 'BMW' is not found in the dictionary.

Similar words:
BMW
aaditya@fedora:~/exp11$ 
```

Explanation

- `grep -q` performs a quiet search for an exact match.
- Suggestions are generated by finding words starting with the first 3 letters.
- Useful for spell checking and learning shell-based text processing.

Result

Successfully implemented shell scripts for sentence splitting, palindrome checking, CSV processing, interactive menus, and dictionary-based word verification.

Conclusion

This experiment strengthened understanding of **string handling, user interaction, text processing, and file manipulation** in Bash scripting.