Assignment 5

a) Types of shell,

- Bourne Shell (sh): The original Unix shell.
- Bash (Bourne Again Shell): A widely used shell on Linux systems, an extended version of the Bourne shell.
- C Shell (csh): A shell with C-like syntax.
- Korn Shell (ksh): Developed by David Korn at AT&T Bell Labs, it incorporates features from both the Bourne shell and the C shell.
- Z Shell (zsh): An extended Bourne shell with many improvements and additional features.

b) Pattern matching,

Wildcard Characters: Used for pattern matching in filenames. Common wildcards are

- * (matches any sequence of characters),
- ? (matches any single character),
- [ ] (matches any character within the brackets).

```
osborn@DESKTOP-GOPGL05:~/msit$ ls *.txt
file1.txt  file2.txt  file3.txt  file4.txt  file5.txt
osborn@DESKTOP-GOPGL05:~/msit$
```

c) Escaping,

Backslash (): Used to escape special characters in shell commands, preventing their special interpretation.

```
osborn@DESKTOP-GOPGL05:~/msit$ echo "Hello\!"
Hello\!
osborn@DESKTOP-GOPGL05:~/msit$
```

d) Quoting,

- Single Quotes ('): Preserves the literal value of all characters enclosed within the quotes.
- Double Quotes (""): Preserves the literal value of all characters enclosed within the quotes, except for the dollar sign ($) and backtick (`) characters.

```
osborn@DESKTOP-GOPGL05:~/msit$ echo 'Hello $USER'
Hello $USER
osborn@DESKTOP-GOPGL05:~/msit$ echo "Hello $USER"
Hello osborn
osborn@DESKTOP-GOPGL05:~/msit$
```

e) Redirection,

- > and >>: Redirects standard output to a file. > overwrites the file, while >> appends to the file.
- <: Redirects standard input from a file.
- 2>: Redirects standard error to a file.

```
osborn@DESKTOP-GOPGL05:~/test$ ls > output.txt
osborn@DESKTOP-GOPGL05:~/test$ ll
total 12
drwxr-xr-x  2 osborn osborn 4096 Apr  3 02:43 ./
drwxr-x--- 12 osborn osborn 4096 Apr  3 02:43 ../
-rw-r--r--  1 osborn osborn   11 Apr  3 02:44 output.txt
osborn@DESKTOP-GOPGL05:~/test$ cat output.txt
output.txt
osborn@DESKTOP-GOPGL05:~/test$ ls >> output.txt
osborn@DESKTOP-GOPGL05:~/test$ cat output.txt
output.txt
output.txt
osborn@DESKTOP-GOPGL05:~/test$ ls
output.txt
osborn@DESKTOP-GOPGL05:~/test$
```

```
osborn@DESKTOP-GOPGL05:~/test$ ls fruits.txt
fruits.txt
osborn@DESKTOP-GOPGL05:~/test$ cat fruits.txt -n
     1  Apple
     2  Banana
     3  Orange
     4  Strawberry
     5  Mango
     6  Pineapple
     7  Watermelon
     8  Grapes
     9  Kiwi
    10  Pear
osborn@DESKTOP-GOPGL05:~/test$ grep "r" < fruits.txt
Orange
Strawberry
Watermelon
Grapes
Pear
```

Redirects error output of the ls command to error.txt

```
osborn@DESKTOP-GOPGL05:~/test$ ls
fruits.txt  output.txt
osborn@DESKTOP-GOPGL05:~/test$ cat file3.txt 2> error.txt
osborn@DESKTOP-GOPGL05:~/test$ ls
error.txt  fruits.txt  output.txt
osborn@DESKTOP-GOPGL05:~/test$ cat error.txt
cat: file3.txt: No such file or directory
osborn@DESKTOP-GOPGL05:~/test$
```

f) Pipe,

| (Pipe): Connects the output of one command to the input of another. Allows for the chaining of commands.

```
osborn@DESKTOP-GOPGL05:~/test$ ls -l | grep ".txt"
-rw-r--r-- 1 osborn osborn 42 Apr  3 02:51 error.txt
-rw-r--r-- 1 osborn osborn 75 Apr  3 02:49 fruits.txt
-rw-r--r-- 1 osborn osborn 22 Apr  3 02:44 output.txt
osborn@DESKTOP-GOPGL05:~/test$
```

g) tee,

tee: Reads from standard input and writes to standard output and files simultaneously.

```
osborn@DESKTOP-GOPGL05:~/test$ ls -l | tee output.txt
total 12
-rw-r--r-- 1 osborn osborn 42 Apr  3 02:51 error.txt
-rw-r--r-- 1 osborn osborn 75 Apr  3 02:49 fruits.txt
-rw-r--r-- 1 osborn osborn 22 Apr  3 02:44 output.txt
osborn@DESKTOP-GOPGL05:~/test$
```

h) Command substitution,

$(command): Executes the command inside the parentheses and replaces it with the command's output.

```
osborn@DESKTOP-GOPGL05:~/test$ echo "Today is $(date)"
Today is Wed Apr  3 02:55:14 IST 2024
osborn@DESKTOP-GOPGL05:~/test$
```

i) Shell variables

- Environment Variables: Variables that are inherited by child processes. Common environment variables include PATH, HOME, USER, etc.
- User-defined Variables: Variables created and used within shell scripts or the shell environment.

```
osborn@DESKTOP-GOPGL05:~/test$ echo $USER
osborn
osborn@DESKTOP-GOPGL05:~/test$ a=20
osborn@DESKTOP-GOPGL05:~/test$ echo $a
20
osborn@DESKTOP-GOPGL05:~/test$
```