Functions

Creating Functions

syntax –

```
function_name () {
   list of commands
}
```

i.e.

```
#!/bin/sh

# Define your function here
ICT () {
   echo "ICT Department"
}

# call your function
ICT
```

Pass Parameters to a Function

i.e.

```
#!/bin/sh

# Define your function here
ICT () {
   echo "ICT Department $a $b $c"
}

# Invoke your function
ICT CS CBA BDA
```

i.e.

```
#!/bin/sh

# Define your function here
ICT () {
   echo "Enter ICT Branch"
   read a
   read b
```

```
   read c
echo "ICT Branch: $a $b $c"
}

# Invoke your function
ICT
```

Nested Functions

i.e.

```
#!/bin/sh

# Calling one function from another
fun_one () {
   echo "ICT Branch"
   fun_two
}

fun_two () {
   echo "CBA CS BDA"
}

# Calling function one.
fun_one
```

The Metacharacters

Unix Shell provides various metacharacters which have special meaning while using them in any Shell Script and causes termination of a word unless quoted.

For example, ? Matches with a single character while listing files in a directory and an ***** matches more than one character. Here is a list of most of the shell special characters (also called metacharacters) −

```
* ? [ ] ' " \ $ ; & ( ) | ^ < > new-line space tab
```

A character may be quoted (i.e., made to stand for itself) by preceding it with a ****.

i.e.

```
#!/bin/sh

echo Hello; Students
```

```
#!/bin/sh

echo Hello\; Studetns
```

Sr.No.	Quoting & Description
1	**Single quote** All special characters between these quotes lose their special meaning.
2	**Double quote** Most special characters between these quotes lose their special meaning with these exceptions − • $ • ` • \$ • \' • \" • \\
3	**Backslash** Any character immediately following the backslash loses its special meaning.
4	**Back quote** Anything in between back quotes would be treated as a command and would be executed.

Double quotes take away the special meaning of all characters except the following −

- **$** for parameter substitution
- Backquotes for command substitution
- **\$** to enable literal dollar signs
- **\`** to enable literal backquotes
- **\"** to enable embedded double quotes
- **** to enable embedded backslashes

```
DATE=`date`

echo "Current Date: $DATE"
```

```
#!/bin/sh

echo "I have \$10"
```

What is Substitution?

The shell performs substitution when it encounters an expression that contains one or more special characters.

Example

Here, the printing value of the variable is substituted by its value. Same time, **"\n"** is substituted by a new line −

```
#!/bin/sh

a=10
echo -e "Value of a is $a \n"
```

You will receive the following result. Here the **-e** option enables the interpretation of backslash escapes.

```
Value of a is 10
```

Following is the result without **-e** option −

```
Value of a is 10\n
```

The following escape sequences which can be used in echo command −

Sr.No.	Escape & Description
1	**** Backslash
2	**\a** alert (BEL)
3	**\b**

	Backspace
4	**\c** suppress trailing newline
5	**\f** form feed
6	**\n** new line
7	**\r** carriage return
8	**\t** horizontal tab
9	**\v** vertical tab