Debugging

For debugging the shell, we can use –v, -x and –n options.
General syntax is as follows:

#sh << options >> << script name >> (OR)
#bash << options >> << script name >>

-x : Display commands and their arguments as they are executed.
-v : Display shell input lines as they are read.
-n : Read commands but do not execute them. This may be used to check a shell script for syntax errors

Bash shell offers debugging options which can be turned on or off using set command as follows.

set -x //set -v //set -n
sum=`expr $1 + $2`
echo $sum
#sh debug.sh 1

```
sum=`expr $1 + $2`
echo $sum

Run the above script
#sh -v degug.sh 1 2
#sh -x degug.sh 1 2
#sh -n degug.sh 1 2
#sh -xv degug.sh 1 2
```

Array Declaration & Initialization:

```
declare -a myArray [OR] myArray=()

myArray[0]="zero"

myArray[1]="one"

myArray[2]="two“

[OR]

myArray=(zero one two)

echo ${myArray[1]}
```

Get user input:

```
read -p 'Enter User Name ' username
read -sp 'Enter the password ' password


#!/bin/bash

# Loop over all arguments

for arg in "$@“
do
  # Use eval to declare the variable
   eval "$arg“
Done

# Now you can use the variables in the script

echo "Name: $name“
echo "Age: $age"
```

STRING MANIPULATION

st="0123456789"

${#string} gives the string length

echo ${#st} → 10

${string:position} extracts sub-string from $string at $position

echo ${st:6} → 6789

${string:position:length} extracts $length characters of sub-string from $string at $position

echo ${st:6:2} → 67

FUNCTION

A function is a collection of statements that execute a specified task. Its main goal is to break down a complicated procedure into simpler subroutines that can subsequently be used to accomplish the more complex routine.

For the following reasons, functions are popular:

Assist with code reuse.
Enhance the program's readability.
Modularize the software.
Allow for easy maintenance.

Function declaration:

```
myFunction() {
   echo "GOOD."
}
```

```
myFunction() {
    echo "Hello $1"
}
```

Calling a function:

```
myFunction
```

```
myFunction Arockia
```

```
#!/bin/bash
# Define a function

add_numbers()
{
 echo $(($1 + $2))
}

# Call the function
add_numbers 5 10
```