Capgemini

UNIX

Lesson 17 : Function ,array and debugging

Lesson Objectives

- At the end of the session you will be able to understand:
  - Functions
  - Array
  - Debugging

Functions in Shell Script

- Use shell functions to modularize the script.
- These are also called as script module
- Normally defined at the beginning of the script.
- Syntax (Function Definition):

```
functionname(){
        commands
        }
```

- Example: Function to create a directory and change directories:
- Use mkcd mydir to call the function. mydir is used as $1 in the function.

```
mkcd()
{
mkdir $1        --$1 is the argument we pass while calling function
cd $1
}
```

Using return statement

Used to come out of a function from within.

- If called *without* an argument, function return value is the same as *exit* status of the last command executed within the function
- If called *with* an argument it returns the argument specified.
- Example:

```
functret()
{
command1
if .....
then
    return 1
else
    return 0
fi
Command2
}
```

Using return statement

```
 Myfunction(){
echo “$*”
echo “The number should be between 1 and 20”
read num
if [ $num –le 1] –a [$num –ge 20]
      return 1;
else
      return 0;
fi
echo “You will never reach to this line”}

echo “Calling the function Myfunction”
if   Myfunction “Enter the number”
then
    echo “The number is within range”
else
   echo the number is out of range”
```

17.2: Arrays
Using arrays

Contains a collection of values accessible by individuals or groups

- Subscript of array element indicates their position in the array.
  - arrayname[subscript]

First element is stored at subscript 0.

- Assign a value in *flowers* array at the first position.
  - Flowers[0]=Rose

Assign values in an array with a single command:

- $ set -A Flowers Rose Lotus

Access individual array elements

- ${arrayname[subscript]}

Using arrays

To print values from array we can use while loop

```
flowers[0]=Rose
 flowers[1]=Lotus
flowers[2]=Mogra
i=0
while [ $i -lt 3 ]
do
echo ${flowers[$i]}
 i=`expr $i+1`
done
```

- **Access all elements:**

```
${array_name[*]}
${array_name[@]}
```

Details

- **Debug a Shell Script :**
- -x option to debug a shell script
- Run a shell script with -x option.
  $ bash -x script-name
  $ bash -x domains.sh
- **set -x** : Display commands and their arguments as they are executed.
- **set -v** : Display shell input lines as they are read.
- You can use above two command in shell script itself:

- *#!/bin/bash*
- **clear**
- *# turn on debug mode*
- **set** -x
- **for** f **in** *
- **do**
- **file** $f
- **done**
- *# turn OFF debug mode*
- **set** +x
- **ls**
- *# **more** commands*

SUMMARY

- In this lesson, you have learnt:
  - ***Function declaration and use***
  - ***Array declaration and use***
  - ***Debugging a script***

Review Questions

- Question 1: Which option we use to debug a script ?
- Question 2: How to display all elements of array ?
- Question 3:In shell programming array holds garbage value if don't set a value?
  - True
  - False