SHELLSCRIPT INTERVIEW QUESTIONS

Mithun Technologies, +91 99809 23226, devopstrainingblr@gmail.com

	Mithun Technologies +91-9980923226	Shell Script Interview Questions info@mithuntechnologies.com	Author	Mithun Technologies	
			Web site	http://mithuntechnologies.com	

Shell Script Test 1

- **What is Shell?**

Ans)

A shell is a program that acts as an interface between a user and the kernel. It allows a user to give commands to the kernel and receive responses from it. Through a shell, we can execute programs and utilities on the kernel. Hence, at its core, a shell is a program used to

execute other programs on our system.

- **How to check how many shells that Linux/Unix server will support? Ans) echo "$SHELL"**

- **What are the different shell types? Ans)**

1. The Bourne Shell (sh)
2. The GNU Bourne-Again Shell (bash)
3. The C Shell (csh)
4. The Korn Shell (ksh)
5. The Z Shell (zsh)

The kernel is the heart of any operating system.
It is responsible for the control management, and execution of processes,

A shell is a program that acts as an interface between a user and the kernel. It allows a user to give commands to the kernel and receive responses from it. Through a shell, we can execute programs and utilities on the kernel. Hence, at its core, a shell is a program used to execute other programs on our system.

1. The Bourne Shell (sh)

The Bourne shell is regarded as the first UNIX shell ever. It is denoted as sh. It gained popularity due to its compact nature and high speeds of operation. However, the Bourne shell has some major drawbacks.
It doesn't have in-built functionality to handle logical and arithmetic operations. Also, unlike most different types of shells in Linux, the Bourne shell cannot recall previously used commands.
The complete path-name for the Bourne shell is /bin/sh and /sbin/sh.

2. The GNU Bourne-Again Shell (bash)

More popularly known as the Bash shell, the GNU Bourne-Again shell was designed to be compatible with the Bourne shell. It incorporates useful features

from different types of shells in Linux such as Korn shell and C shell.
It allows us to automatically recall previously used commands and edit them with help of arrow keys, unlike the Bourne shell.
The complete path-name for the GNU Bourne-Again shell is /bin/bash.

3. The C Shell (csh)

The C shell was created at the University of California by Bill Joy. It is denoted as csh. It was developed to include useful programming features like in-built support for arithmetic operations and a syntax similar to the C programming language.
Further, it incorporated command history which was missing in different types of shells in Linux like the Bourne shell. Another prominent feature of a C shell is "aliases".
The complete path-name for the C shell is /bin/csh.

4. The Korn Shell (ksh)

The Korn shell was developed at AT&T Bell Labs by David Korn, to improve the Bourne shell. It is denoted as ksh. The Korn shell is essentially a superset of the Bourne shell.

Besides supporting everything that would be supported by the Bourne shell, it provides users with new functionalities. It allows in-built support for arithmetic operations while offereing interactive features which are similar to the C shell.
The Korn shell runs scripts made for the Bourne shell, while offering string, array and function manipulation similar to the C programming language. It also supports scripts which were written for the C shell. Further, it is faster than most different types of shells in Linux, including the C shell.
The complete path-name for the Korn shell is /bin/ksh.

5. The Z Shell (zsh)

The Z Shell or zsh is a sh shell extension with tons of improvements for customization. If you want a modern shell that has all the features a much more, the zsh shell is what you're looking for.
Some noteworthy features of the z shell include:
Generate filenames based on given conditions
Plugins and theming support
Index of built-in functions
Command completion

Shell	Complete path-name	Prompt for root user	Prompt for non root user

Bourne shell (sh)	/bin/sh and /sbin/sh	#	$
GNU Bourne-Again shell (bash)	/bin/bash	bash-VersionNumber#	bash-VersionNumber$
C shell (csh)	/bin/csh	#	%
Korn shell (ksh)	/bin/ksh	#	$
Z Shell (zsh)	/bin/zsh	<hostname>#	<hostname>%

- **How many types of comments does shell script support? Ans)**

Bash script provides support for two types of comments, just like the other programming language.
Single Line Comment
Multiple/ Multi-Line Comment

#This is a single-line comment in Bash Script.

<<COMMENTS
This is the first comment
This is the second comment
This is the third comment
COMMENTS

*** What is shebang in shell scripting?**

So, if the first line of a script is:
#!/bin/bash
It means the interpreter should be bash shell. If the first line is:
#!/bin/zsh
It means the interpreter to be used is Z shell.

- **What is command line arguments? Ans)**

During shell script execution, values passing through command prompt is called as

command line arguments.

For example, while running a shell script, we can specify the command line arguments as "sh scriptfile.sh arg1 arg2 arg3"

While using command line arguments follow the below important points.

We can specify n number of arguments, there is no limitation.
Each argument is separated by space.

- **What is the difference between $* and $@ ? Ans)**

$* Stores all the arguments that were entered on the command line ($1 $2 ...). "$@" Stores all the arguments that were entered on the command line, individually quoted ("$1" "$2"

$* and $@

- The difference between $* and $@
  - $* : all arguments are formed into a long string
  - $@ : all arguments are formed into separated strings
- Examples: test.sh

```
for i in "$*" ;  do
    echo $i
done

% test.sh 1 2 3
1 2 3
```

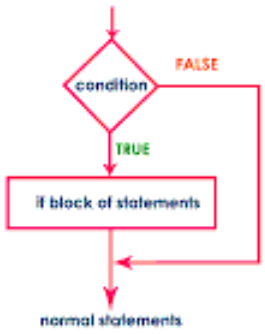
```
for i in "$@" ;  do
    echo $i
done

% test.sh 1 2 3
1
2
3
```

- **Write down the syntax for if condition? Ans)**

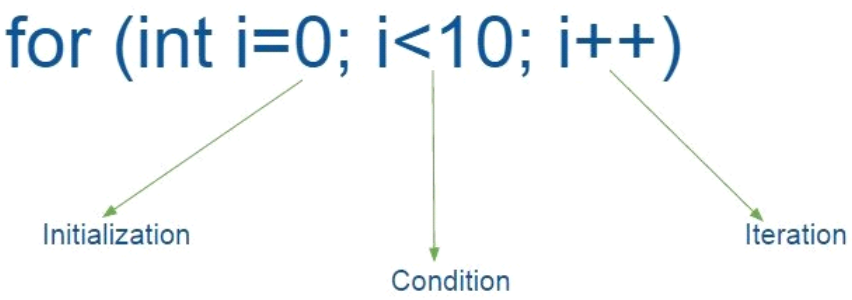
Syntax

```
if ( condition )
{
  .....
  block of statements;
  .....
}
```

Execution flow diagram



- **Write down the syntax for for loop? Ans)**



- **Write down the syntax for function and write one function and call that function? Ans)**

```
greetfn(){

echo "Welcome to Shell script! "

}

echo "Calling greetfn() "


greetfn
```

Writing one function

```
#!/bin/sh

function example {
echo "Hello Learner"

}
```

- **What is the difference between > and >> and < and what is the standard output and standard error codes?**

Ans)

Using shell scripts, we can redirect - the output of a command to a file or - redirect an output file as an input to other commands.

In Shell script there are mainly 3 types of redirect symbols as follows.
1. > Redirect standard output Example:
ls > ls-file.txt

The above command will redirect the output of the " ls " to the file " ls-file ".
If the file " ls-file " already exist, it will be overwritten. Here you will loose the existing data.
>> Append standard output Example:
date >> ls-file.txt

The output of the date command will be appended to the file " ls-file ".
In this case you will not loose any data. The new data gets added to the end of the file.
< Redirect standard input Example:
cat < ls-file.txt

This redirection symbol takes input from a file.
In the above example the cat command takes the input from the file " ls-file " and displays the "ls-file" content.

- **How to display one variable (take variable name as technology)value? Ans)**

- **How many types of variables in Shell scripting? Ans)**

There are two types of variables in Linux shell script. Those are 1) System variables 2) User defined variables (UDV).

- **Write some System defined variables? Ans)**

Created and maintained by Linux bash shell itself. This type of variable is defined in CAPITAL LETTERS.
There are many shell inbuilt variables which are used for administration and writing shell scripts. To see all system variables, type the following command at a console / terminal:
env or printenv

Use echo command to display variable value as follows.
File name : system_variables.sh

```
#! /bin/bash
echo 'BASH='$BASH
echo
'BASH_VERSION='$BASH_VE
RSION echo 'HOSTNAME='
$HOSTNAME
echo 'TERM='$TERM echo
'SHELL='$SHELL
echo 'HISTSIZE='$HISTSIZE
echo
'SSH_CLIENT='$SSH_CLIENT
echo 'QTDIR='$QTDIR
echo 'QTINC='$QTINC
echo 'SSH_TTY='$SSH_TTY
echo 'RE_HOME='$JRE_HOME
echo 'USER='$USER
```

- **What is String?**

Ans)

Strings are scalar and there is no limit to the size of a string.
Any characters, symbols, or words can be used to make up your string.

- **How to find the length of the given string? Ans)**

Create a bash file named “len1.sh” and add the following script. Here, a string value will be taken from the user and the length of the string value will be counted by using `expr` command that will be printed later.

len1.sh

```
#!/bin/bash
echo  “Enter a string:”
read strval
len=`expr "$strval" : '.*'`
echo "The length of the input string is $len"
```

Run the script.

```
$ bash len1.sh
```

Output:

Here, “I like Programming” is taken as input and the length of the string is 18.

Shell Script Test 2

- **Write a shell script to accept the name from name and age from the user and display that back to the user.**

Ans)

```
echo "enter your name"
read name
echo "enter your age"
read age
echo "your name is" = $name
echo "your age is" = $age
```

- **Write a shell script to accept a file name from the user and make a copy of that file. Ans)**

- **Write a shell script to accept file name from the user and display the contents of file. If the file doesn't exist then try curbing the error and display a user friendly error to user. Ans)**

- **Write a shell script to accept a file name from user and check whether its an ordinary file or a directory. In case of file show the contents of file and if itís a directory show number of files in that directory.**

Ans)

- **Write a shell script to accept a file name from user. Check whether file has all the permissions if not assign the respective permissions to that file.**

Ans)

SHELL CONDITIONAL EXPRESSIONS

You can use conditional expressions to find out file permissions. These are used by the [[compound command and the test and [builtin commands to test file attributes and perform string and arithmetic comparisons.

Conditional Expression, Meaning

-a file, True if file exists.
-b file, True if file exists and is a block special file.
-c file, True if file exists and is a character special file.
-d file, True if file exists and is a directory.
-e file, True if file exists.
-f file, True if file exists and is a regular file.
-g file, True if file exists and is set-group-id.
-h file, True if file exists and is a symbolic link.
-k file, True if file exists and its "sticky" bit is set.
-p file, True if file exists and is a named pipe (FIFO).
-r file, True if file exists and is readable.
-s file, True if file exists and has a size greater than zero.
-t fd, True if file descriptor fd is open and refers to a terminal.
-u file, True if file exists and its set-user-id bit is set.
-w file, True if file exists and is writable.
-x file, True if file exists and is executable.
-O file, True if file exists and is owned by the effective user id.
-G file, True if file exists and is owned by the effective group id.
-L file, True if file exists and is a symbolic link.
-S file, True if file exists and is a socket.
-N file, True if file exists and has been modified since it was last read.

-f $file -> returns true if file exists.
-r $file -> returns true if file has Read permission
-w $file -> returns true if file ha write permission.
-x $file -> returns true if file has Executed permission.
-a -> it is used for checking multiple conditions, same as && operator.

```
echo "enter filename :"
read filename
if [ -r $filename ]
then
```

```
echo "file has read permisions"
else
echo "do not have read permissions"
fi
```

- **Write a shell script to accept a file name from the user and sort the file. If the file doesn't exist curb the error message and show the user-friendly message.**

Ans)

Shell Script Test 3

devopstrainingblr@gmail.com

- **Write down the syntax for if condition? Ans)**

```
echo  "Enter the name of the file:"
read filename
if [ -f $filename ]
   then
echo "$filename is existed..."
cat $filename
else
echo "$filename not existed"
touch $filename
fi
```

- **Write down the syntax for "for loop"? Ans)**

```
echo "For loop demo started"
for (( i=1; i<=5; i++ ))
do
  echo $i
  echo ""
done
echo "For loop demo over"
```

- **Write down the syntax for function and write one function and call that function? Ans)**

- **How many types of variables in Shell scripting? Ans)**

A shell script has two types of variables :
System-defined variables are created/defined by the Operating System(Linux) itself. These variables are generally defined in Capital Letters and can be viewed by "set" command.
User-defined variables are created or defined by system users and the values of variables can be viewed by using the command "echo".

devopstrainingblr@gmail.co

m

- **Write a shell script to accept a file name from the user and sort the file. If the file doesn't exist curb the error message and show the user-friendly message.**

Ans)

- **How can you run a shell script in debug mode? --> CricBuzz Ans)**

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

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

sh -x add  3 4
++ expr 3 + 4
+ sum=7
+ echo 7
7

sh -v add  3 4
sum=`expr $1 + $2`
```

echo $sum
7

- **How to run a shell script in background? Ans)**

command &

Ping

- **How to send error logs and stdout logs in different files? Ans)**

Input , output redirection as above

- **How to run a script at boot level? Ans)**

- **In Shell script what are the predefined variables are there? Ans)**

- **How to declare a variable in a shell script? Ans)**

To declare a variable, just type the name you want and set its value using the equals sign (=).
Check out this example:

```
#!/bin/bash
name="Mokhtar"
age=35
total=16.5
echo $name  #prints Mokhtar
echo $age   #prints 35
echo $total #prints 16.5 Copy
```

As you can see, to print the variable's value, you should use the dollar sign ($) before it.
Note that there are no spaces between the variable name and the equals sign, or between the equals sign and the value.
If you forget and type a space in between, the shell will treat the variable as if

it were a command, and, since there is no such command, it will show an error.

28)What is $?, $#, $* ?
Ans)

$# Stores the number of command-line arguments that were passed to the shell program.

$? Stores the exit value of the last command that was executed.

$0 Stores the first word of the entered command (the name of the shell program).

$* Stores all the arguments that were entered on the command line ($1 $2 ...).

"$@" Stores all the arguments that were entered on the command line, individually quoted ("$1" "$2" ...).

So basically, $# is a number of arguments given when your script was executed. $* is a string containing all arguments. For example, $1 is the first argument and so on. This is useful, if you want to access a specific argument in

your script.
As Brian commented, here is a simple example. If you run following command:

command -yes -no /home/username

$# = 3
$* = -yes -no /home/username
$@ = array: {"-yes", "-no", "/home/username"}
$0 = command

```
for ARG in $*
do
echo $ARG
done
~
```

1 2 3 4 5 6

```
for ARG in $#
do
echo $ARG
done
~
6
```

- **What is the command for status of last command execution or last command execution status in linux and what is the output of that command?**

Ans)

"$?" is a variable that holds the return value of the last executed command. "echo $?" displays 0 if the last command has been successfully executed and displays a non-zero value if some error has occurred.

- **What is the purpose of export**

command? Ans)

The export command is a built-in utility of Linux Bash shell. It is used to ensure the environment variables and functions to be passed to child processes. It does not affect the existing environment variable. Environment variables are set when we open a new shell session.

- **How to read a command line input in shell script? Ans)**

By using Read command

Dell

- **How to find the previous command execution status? Ans)**

Echo $?

- **How to perform the arithmetic operations in Shell script? Ans)**

We use the keyword " expr " to perform arithmetic operations

devopstrainingblr@gmail.co

m

+91-9980923226

- **Write a script (in any scripting language) to find the number of occurrences of the word "Failure" in a file sample.txt and replace all occurrence of "Failure" to "Success".**
Ans)

- **Write a script (in any scripting language) to input "n" number of username as parameters and to find it that user is present or not. Output should be "Username - Present" or "Username - Not Present"**
Example: CheckUser.sh User1, User2,User3,. .. N
Output : User 1 - Present User 2 - Not Present User 3 - Logged Ans)

- **Write a single Bash or Python or Perl script to check file system usage on Linux host. send an email to the abc@xyz.com. If one or more file system usage is above the threshold.(Ex.85%) - PWC – Written Test.**
Ans)

- **Write a bash or Python or PERL script to list the unique shells in /etc/password file and print it string with the number of users using that shell.(Note: shell is the last filed in /etc/passwd file delimited by ":") --> PWC**
Ans)

- **There are multiple sub directories under a directory. Ex "/var/tmp/app-code" on a linux host. Those sub directories cobtain one or more XML files. Some of those XML files have an IP address (EX: 85.) hardcoded.Write a one liner bash command to replace all the occurrences of this IP address wit a variable EX. SIP_Address in all the files it is existing. --> PWC**
Ans)

- **Write a single Bash or Python or Perl script to check file system usage on Linux host.send an email to the abc@xyz.com .If one or more file system usage is above the threshold.(Ex.85%)**
What is first line written in shell script? What is the meaning of that? If I didn't write that line what will happen? --> PWC
Ans)

- **What is the difference between /etc/hosts and /etc/resolve.conf? What are the types of DNS records? --> PWC**
Ans)

What is DNS and why is it used?

DNS (Domain Name System) is mechanism to make internet human friendly. Computer communicate with each other using their IP addresses. There are lots of IP address in internet world, and It is very difficult to remember all IP address . To solve this issue and make more human friendly DNS invented. IP address map with host-name in DNS server.
When we enter a domain name into our browser like www.goole.com computer find our nearest DNS server and ask what is the correct IP address for www.google.com. Than it returns the IP address to our system to communicate with www.google.com.

/etc/hosts file
/etc/hosts file is the most important file in Linux operating system. It is a text file for name resolution.
 # cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.0.2 client01

192.168.0.3 client02 www.example.com
Both fields are separated by space or tab followed by IP and Host-name per line. Host-name contain only alphanumeric characters, minus sign (-) and period (.) .
Fields Explanations :
192.168.0.3 : IP address
client02 : Host-name
www.example.com : Aliases of IP address
You can also see other entries in /etc/hosts file like 127.0.0.1 is loopback addresses. Which is pointed to "localhost" host-name.

/etc/resolv.conf file

This is the another important file in Linux operating system. It contains information that help to computer to convert domain name to its IP address . All process called resolving.

 # cat /etc/resolv.conf

nameserver 173.204.4.5

nameserver 173.204.4.7

Above you can see "nameserver" directive which is pointed to the IP address of Name Server.

- **How to write a script when the first command is executed successfully then execute another command?**

Ans)

```
java -
version if
[ $? -eq
0 ] then
echo "Previous command executed successfully"
else
echo "Previous command executed not successfully
" fi
```

- **Can you tell me the syntax for forloop and while loop? Ans)**

```
a=0
```

```
while
[ $a -lt 10 ]
do
    echo $a
    a=`expr $a + 1`
Done


echo "For loop demo started"
for (( i=1; i<=5; i++ ))
do
   echo $i
   echo ""
done
echo "For loop demo over"
```

- **How to print shell name? Ans)**

```
Echo $SHELL or echo
$0
```

- **How to assign a all the arguments to a single variable? Ans)**

Assigning the arguments to a regular variable (as in args="$@") mashes all the arguments together like "$*" does. If you want to store the arguments in a variable, use an array with args=("$@") (the parentheses make it an array), and then reference them as e.g. "${args[0]}" etc.

- **How to print the current process id of current shell? Ans)**

There is a special variable called "$" and "$BASHPID" which stores the process ID of the current shell. Go ahead and run the below

command to see what is the process ID of your current shell. Both "$" and "$BASHPID" is going to return the same value.

Echo $BASHPID – prints id

- **How to know the file that is entering randomly to my script? Ans)**

- **How to divide two variables in shell script? Ans)**

Division of two numbers using expr in shell script

```
#Unix shell script for division of two numbers
#using expr

num1=100
num2=20

ans=`expr $num1 /  $num2`

echo $ans
```

Output
5

- **What is trap?**

Ans)

A built-in bash command that is used to execute a command when the shell receives any signal is called `trap`. When any event occurs then bash sends the notification by any signal. Many signals are available in bash. The most common signal of bash is SIGINT (Signal Interrupt). When the user presses CTRL+C to interrupt any process from the terminal then this signal is sent to notify the system. How you can use trap command to handle different types

of signals is explained in this tutorial.

-l, It is used to display the list of all signal names with corresponding number.
-p, It is used to display signal command or trap command for signal_spec.
arg, It is used to execute a command when the shell receives the signal(s).
signal_spec, It contains signal name or signal number.

- **What is shift in shell script? Ans)**

The shift command is used to move the command line arguments one position left. The first argument is lost when you use the shift command. Shifting command line arguments is useful when you perform a similar action to all arguments, one-by-one, without changing the variable name.

- **How to run our script in background? Ans)**

Running shell command or script in background using nohup command.

- **How to know the running back ground process id ? Ans)**

```
ps -aux
```

- **How to print only directories? Ans)**

Ls –d*

devopstrainingblr@gmail.com

m

- **How to print the directory only started with number? Ans)**

- **How to grep two strings at a time? Ans)**

- **How to grep a string that is started with some string and ends with some string like a…..b ? Ans)**

- **How to print string that starts with a? Ans)**

- **Did you work on arrays? Ans)**

Arrays provide a

method of grouping a set of variables. Instead of creating a new name for each variable that is required, you can use a single array variable that stores all the other variables.

- **How will you give access of your script to a particular user? Ans)**

Chmod command

chmod 754 myfile

4 stands for "read",
2 stands for "write",
1 stands for "execute", and
0 stands for "no permission."

Search Foobar

Write a Bash script that searches all .c files in the current directory (and its sub-directories recursively) for occurrences of the word "foobar" . Your search should be case-sensitive(that applies both to filenames and the word "foobar").

Note that an occurrence of "foobar" only counts as a word if it is either at the beginning of the line or preceded by a non-word-constituent character or similar if it is either at the end of the line or followed by a non-word constituent character. Word constituent characters are letters, digits and underscores. For instance, "int a + foobar()", "#include <foobar.h>"and "foobar*10"are valid occurrence of the word "foobar", while "foobar1000", "foobar", "foobar_"and "1foobaria"are not.

Your script should list all the valid occurrences on the standard output, in the following format:

```
[file_path]:[line_number]:[line_content]
```

If there are multiple occurrence in one line, list the line only once. The order of lines does not matter, they can be sorted in any way.

In order to succeed your script must have exit status equal to 0. An easy way to ensure that is to end it with "; true".

The directory being searched may contain various files with different extensions (or no extension at all).

Example 1
The current directory is as follows:

```
header.h
main.c
bashrc
external_lib
      |--foobar.c
      |--foobar.h
```


devopstrainingblr@gmail.com

The files have following content:

header.h:

```
int foobar();
```

main.c:

```
#include "header.h"
#include "external_lib/foobar.h"
int main(){
     return foobar() + foobar(); }
```

external_lib/foobar.c:

```
#include "foobar.h"
int foobar() { return 0; }
const char* foobarwoof() { return "WOOF!"; }
```

external_lib/foobar.h:

```
int foobar();
```

bashrc:

```
CXXFLAGS += foobar
```

Your script should write to standard output the following content:

```
./main.c:2:#include "external_lib/foobar.h"
./main.c:4:    return foobar() + foobar();
./external_lib/foobar.c:1:#include "foobar.h"
./external_lib/foobar.c:2:int foobar() { return 0;}
```

Example 2

There is only one file in the current directory, name main.c:

```
int foobar(int n) {
     if (n ==0) return 1;
     if (n ==1) return 1;
     return foobar(n-1) + foobar(n-2);
```

Your script should write to standard output the following content:

```
./main.c:1 int foobar(int n) {
./main.c:4      return foobar(n-1) + foobar(n-2);
```

	Mithun Technologies +91-9980923226	**Shell Script Interview Questions info@mithuntechnologies.com**	**Author**	**Mithun Technologies**	
			Web site	**http://mithuntechnologies.com**	

Example 3

The directory looks as follows:

```
foo.c
foobar.cx
main.cc
dir.c/ (empty directory)
```

The files have the following content:

foo.c

```
int Foobar(int _foobar_);
```

main.cc:

```
const int foobar = 0;
int main() {
      return foobar;
}
```

In this example , your script should write nothing

Basic Coding Skills

A non-empty array A consisting of N integers is given.

Array A represents a linked list. A list is constructed from this array as follows:

- The first node (the head) is located at the index 0;
- The value of a node located at index K is A[K]
- If the value of a node is -1 then it is the last node of the list
- Otherwise, the successor of a node located at index K is located at index A[K] (you can assume that A[K] is a valid index , that is 0 < A[K] < N).

For example for array A such that :

```
A[0] = 1
A[1] = 4
A[2] = -1
A[3] = 3
A[4] = 2
```

The following list is constructed :

- The first node (the head) is located at index 0 and has a value of 1;
- The second node is located at index 1 and has a value of 4
- The third node is located at index 4 and has a value of 2;
- The fourth node is located at index 2 and has a value of -1.

	Mithun Technologies +91-9980923226	Shell Script Interview Questions info@mithuntechnologies.com	Author	Mithun Technologies	
			Web site	http://mithuntechnologies.com	