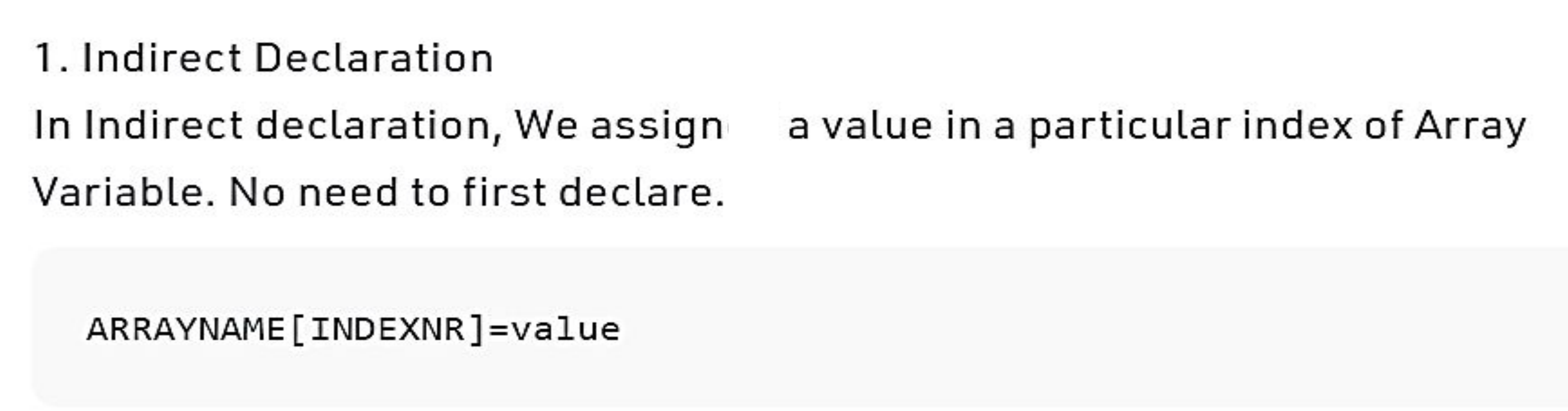
OS Lab 5- Shell Scripting (Arrays)

Arrays

- An array is a systematic arrangement of the same type of data.
- In Shell script Array is a variable
  - which contains multiple values may be of same type
  - or different type
  - An array is zero-based i.e., indexing start with 0.

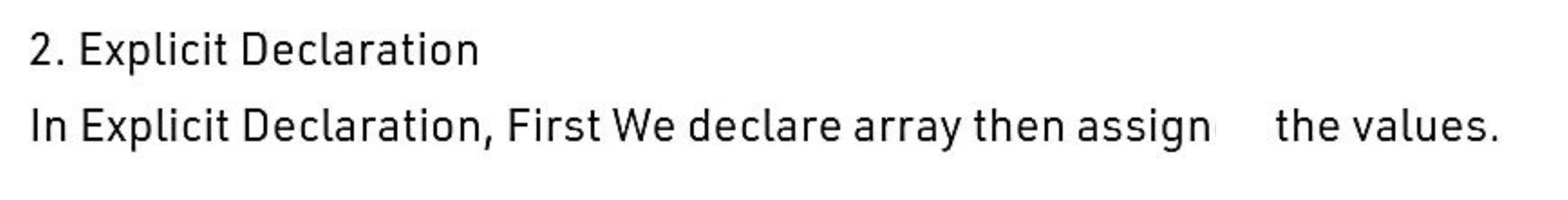
Declaring an array



1. Indirect Declaration

In Indirect declaration, We assign a value in a particular index of Array Variable. No need to first declare.

```
ARRAYNAME[INDEXNR]=value
```

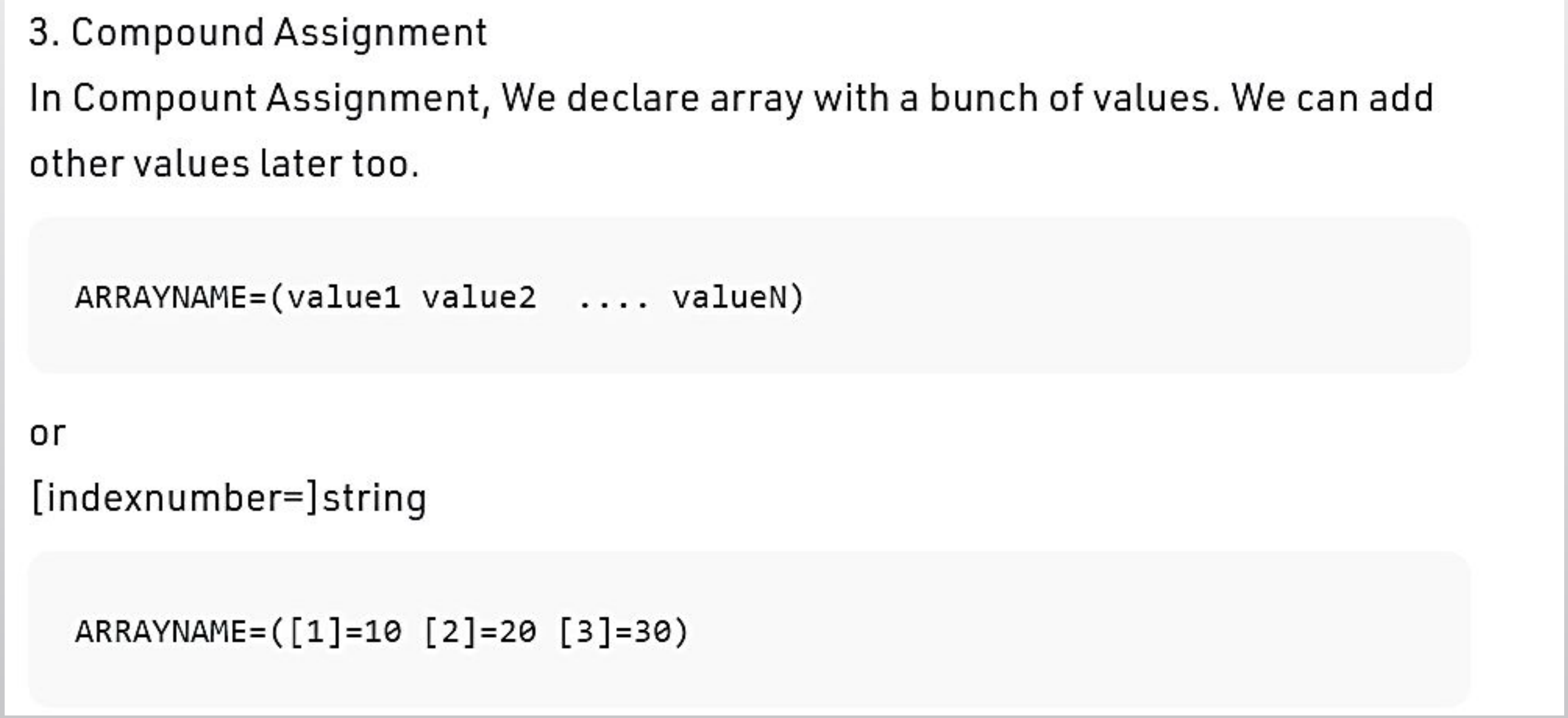


2. Explicit Declaration

In Explicit Declaration, First We declare array then assign the values.

```
declare -a ARRAYNAME
```

Declaring an array



3. Compound Assignment

In Compount Assignment, We declare array with a bunch of values. We can add other values later too.

```
ARRAYNAME=(value1 value2  .... valueN)
```

or

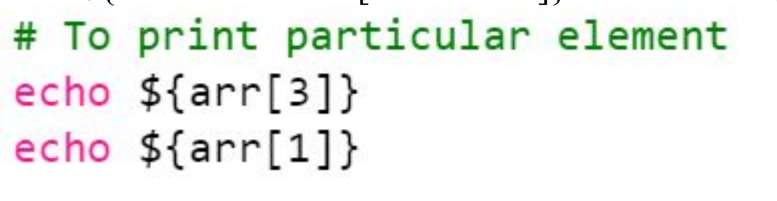
[indexnumber=]string

```
ARRAYNAME=([1]=10 [2]=20 [3]=30)
```

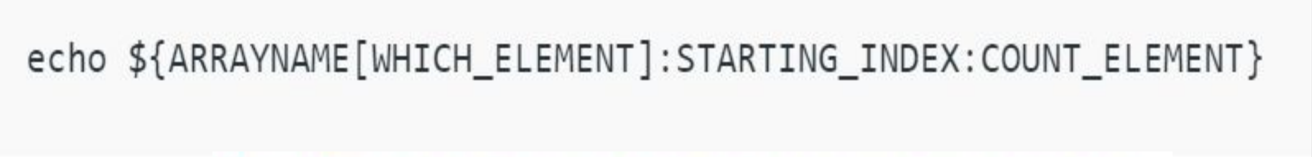
Printing the value of arrays

To Print Selected index element

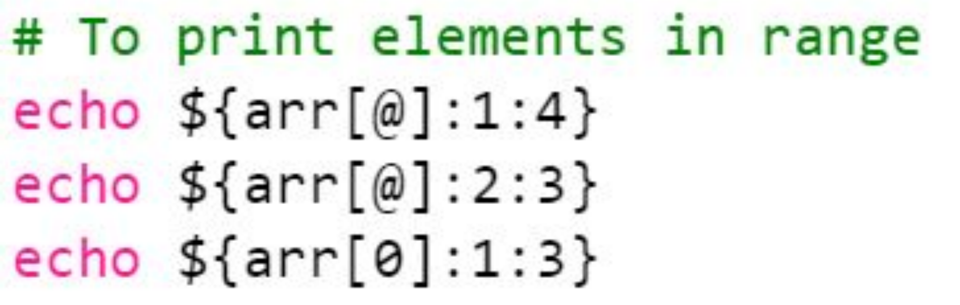
echo ${ARRAYNAME[INDEXNR]}



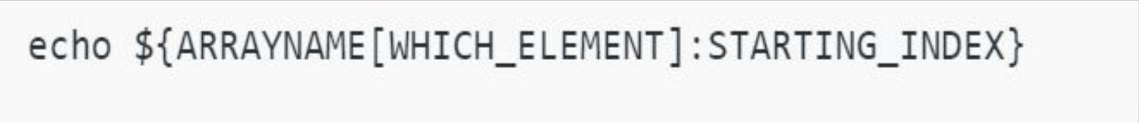
```
# To print particular element
echo ${arr[3]}
echo ${arr[1]}
```



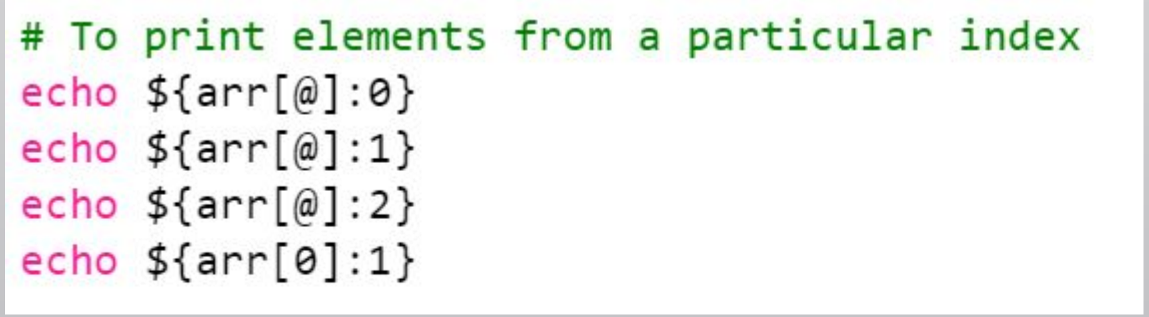
```
echo ${ARRAYNAME[WHICH_ELEMENT]:STARTING_INDEX:COUNT_ELEMENT}
```



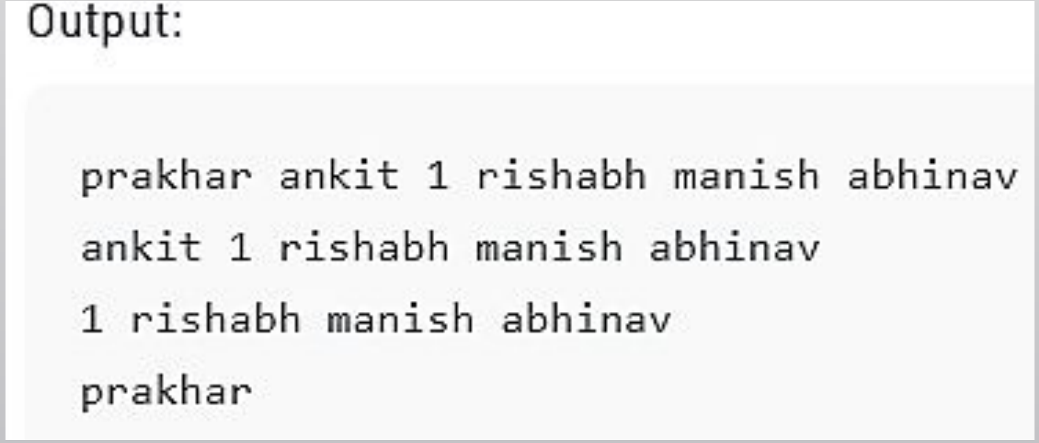
```
# To print elements in range
echo ${arr[@]:1:4}
echo ${arr[@]:2:3}
echo ${arr[0]:1:3}
```



```
echo ${ARRAYNAME[WHICH_ELEMENT]:STARTING_INDEX}
```



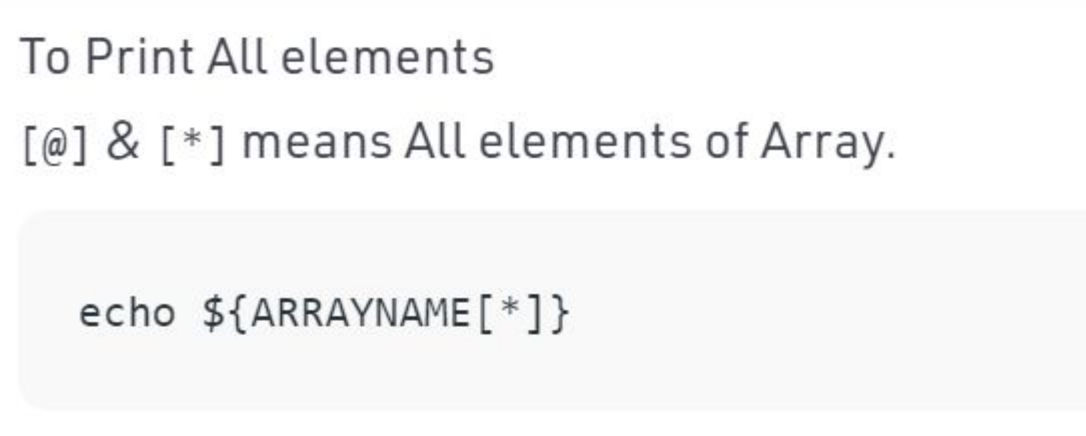
```
# To print elements from a particular index
echo ${arr[@]:0}
echo ${arr[@]:1}
echo ${arr[@]:2}
echo ${arr[0]:1}
```



Output:

```
prakhar ankit 1 rishabh manish abhinav
ankit 1 rishabh manish abhinav
1 rishabh manish abhinav
prakhar
```

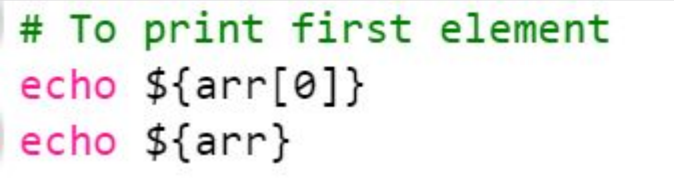
Printing the value of arrays



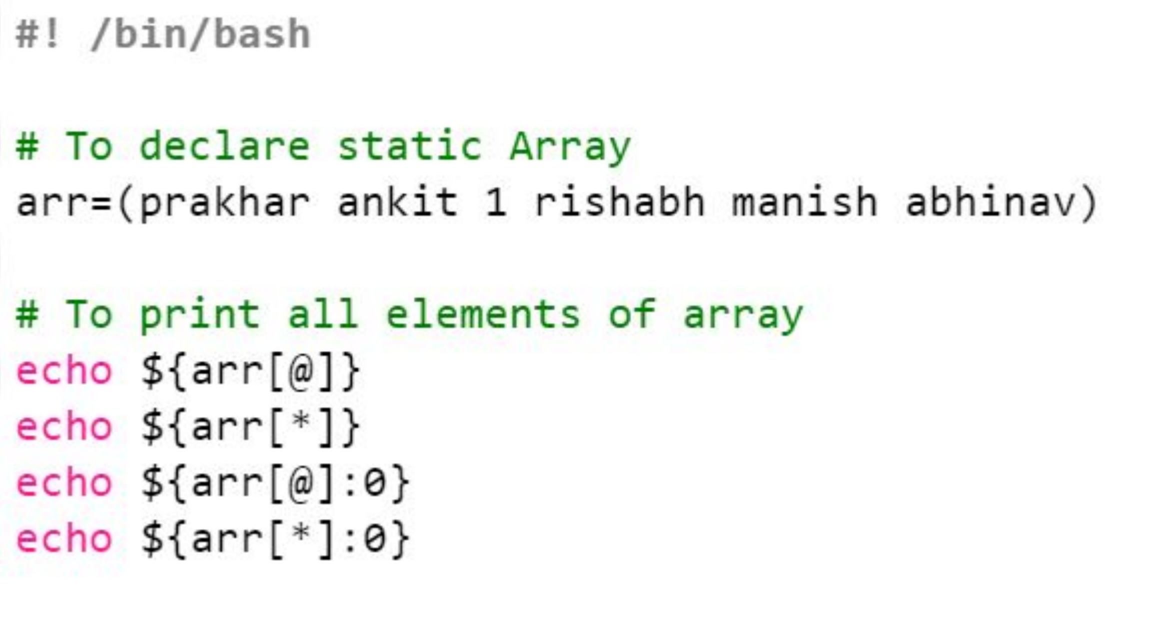
To Print All elements

[@] & [*] means All elements of Array.

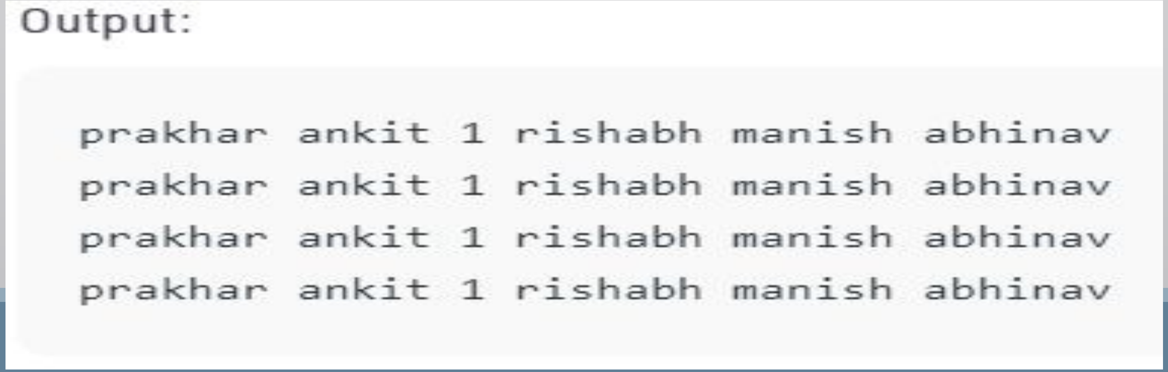
```
echo ${ARRAYNAME[*]}
```



```
# To print first element
echo ${arr[0]}
echo ${arr}
```



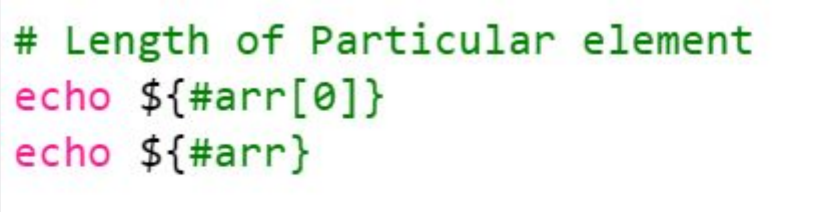
```
#! /bin/bash

# To declare static Array
arr=(prakhar ankit 1 rishabh manish abhinav)

# To print all elements of array
echo ${arr[@]}
echo ${arr[*]}
echo ${arr[@]:0}
echo ${arr[*]:0}
```



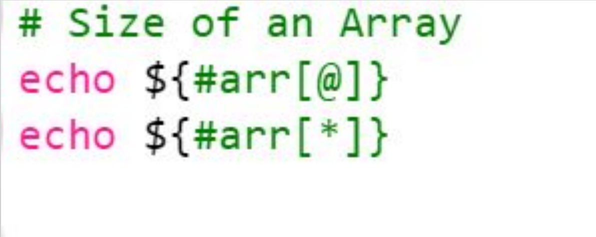
Output:

```
prakhar ankit 1 rishabh manish abhinav
prakhar ankit 1 rishabh manish abhinav
prakhar ankit 1 rishabh manish abhinav
prakhar ankit 1 rishabh manish abhinav
```

Find length of an array and its element

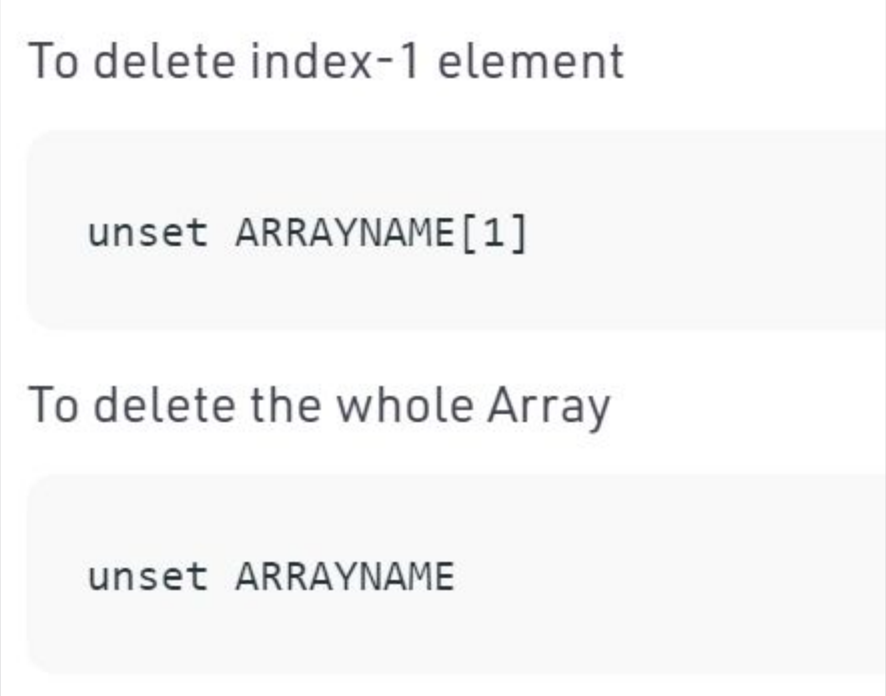


```
# Length of Particular element
echo ${#arr[0]}
echo ${#arr}
```



```
# Size of an Array
echo ${#arr[@]}
echo ${#arr[*]}
```

Deletion



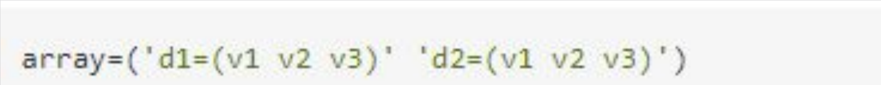
To delete index-1 element

```
unset ARRAYNAME[1]
```

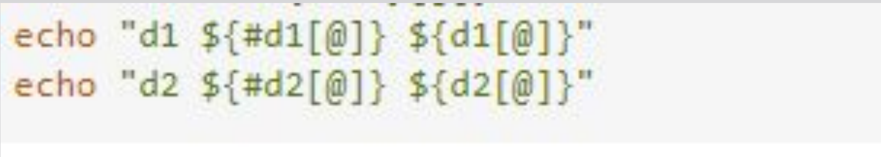
To delete the whole Array

```
unset ARRAYNAME
```

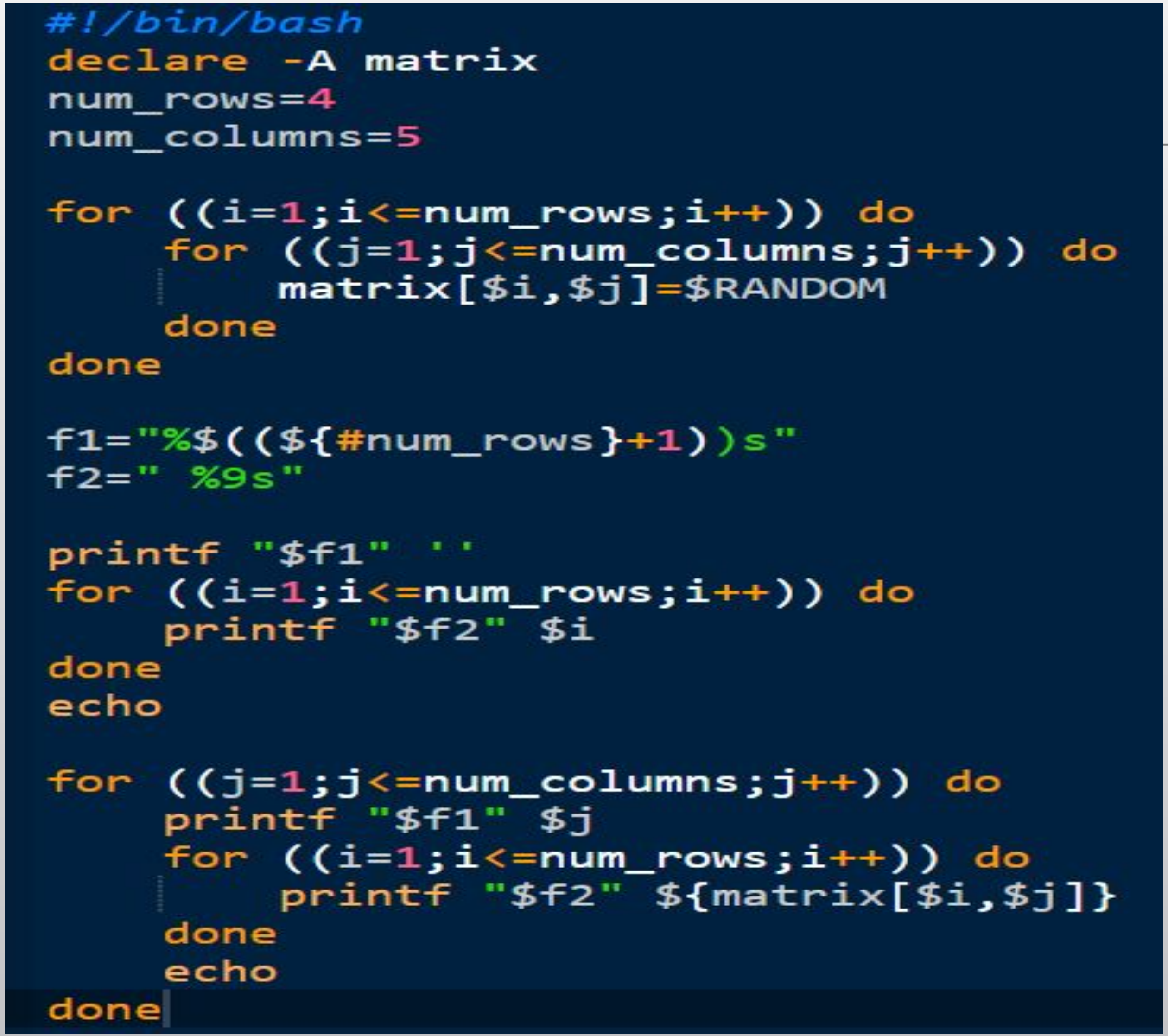
Multi-dimensional Arrays

- **Bash does not support multi-dimensional arrays**
- But we can create associative array and you need to explicitly declare that the variable (array) is an associative array.
  - Declare –A arrayname
- 

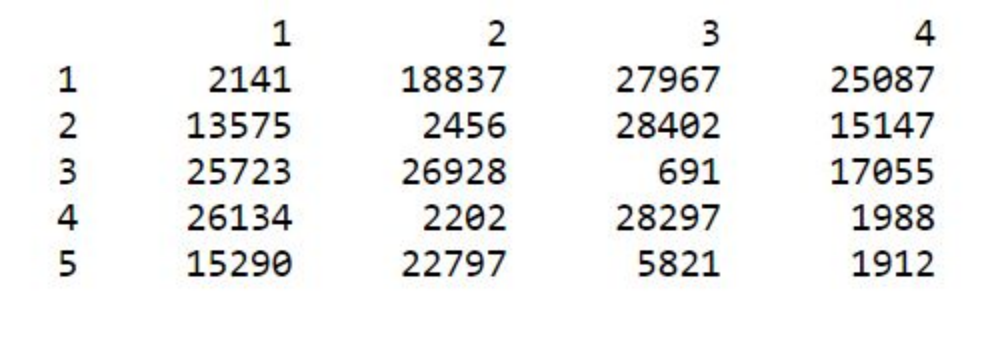
```
array=('d1=(v1 v2 v3)' 'd2=(v1 v2 v3)')
```



```
echo "d1 ${#d1[@]} ${d1[@]}"
echo "d2 ${#d2[@]} ${d2[@]}"
```



```bash
#!/bin/bash
declare -A matrix
num_rows=4
num_columns=5

for ((i=1;i<=num_rows;i++)) do
    for ((j=1;j<=num_columns;j++)) do
        matrix[$i,$j]=$RANDOM
    done
done

f1="%$((${#num_rows}+1))s"
f2=" %9s"

printf "$f1" ''
for ((i=1;i<=num_rows;i++)) do
    printf "$f2" $i
done
echo

for ((j=1;j<=num_columns;j++)) do
    printf "$f1" $j
    for ((i=1;i<=num_rows;i++)) do
        printf "$f2" ${matrix[$i,$j]}
    done
    echo
done
```



```
           1         2         3         4
 1      2141     18837     27967     25087
 2     13575      2456     28402     15147
 3     25723     26928       691     17055
 4     26134      2202     28297      1988
 5     15290     22797      5821      1912
```

Managing strings

```
#!/bin/bash
# Basic for loop
declare -a names=("Stan kil" "Kyle" "Cartman")
for name in "${names[@]}"
do
echo $name
done
echo All done
```