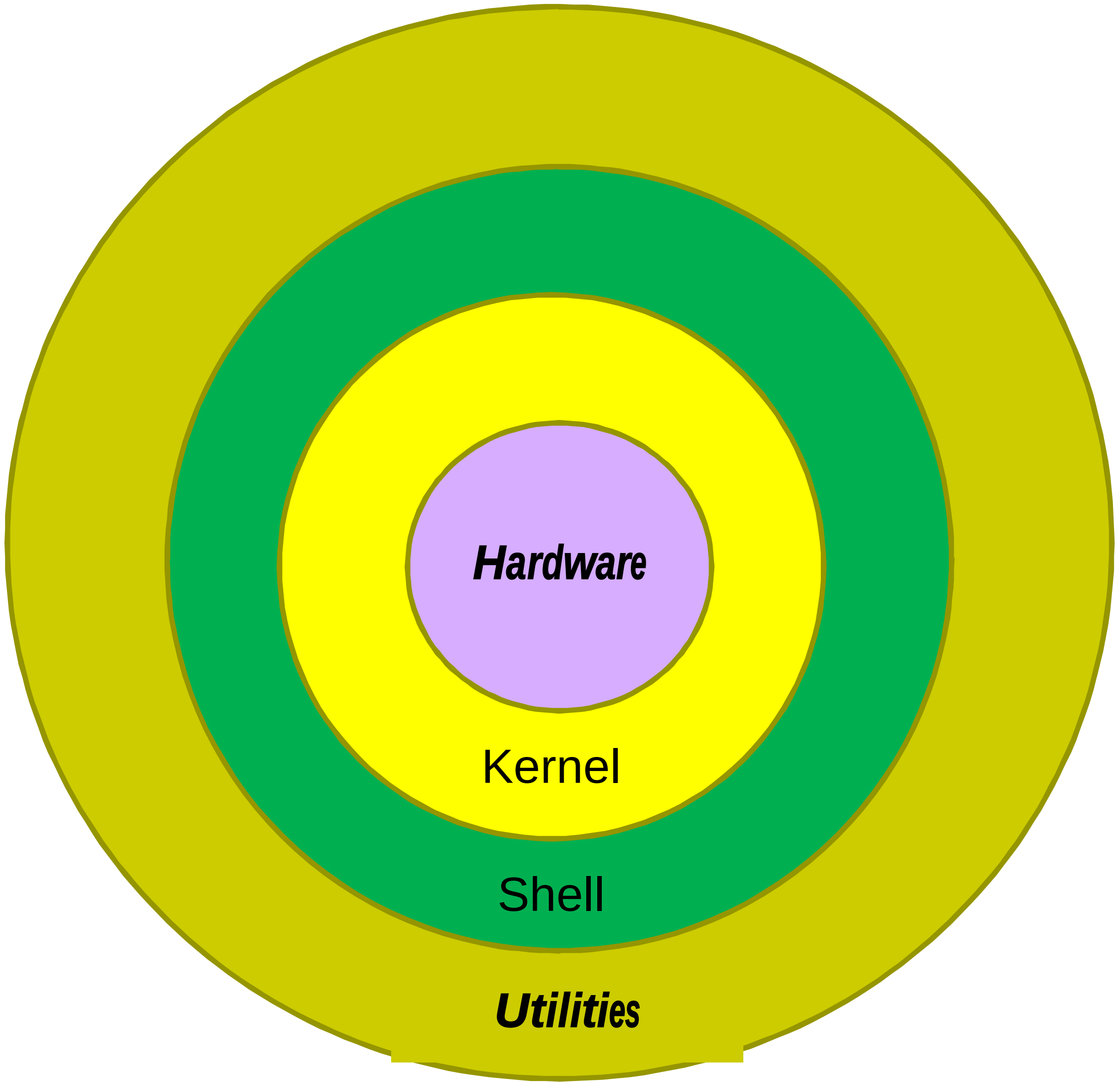
Shell Programming

Dr. Padmaja Joshi


Hardware
Kernel
Shell
Utilities

What is Shell Scripting?

- *Script written for a shell or command line interpreter of an OS*
- *Different Command Line Interfaces - Unix Shell, Windows PowerShell , MS-DOS command.com*
- *Whatever that can be entered at the shell can be grouped together and written in a shell script like batch jobs*
- *Different Unix shells – Bourne (known as sh; Stefen Bourne AT&T labs), bash (Bourne again), csh (Bill Joy @ Berkley), tcsh, ksh (David Korn)*

Shells

C

Korn

Borne

GNU

Bash

- *Know your shell*

```
$ echo $SHELL
```

Comments and Variables

- *Comments in Shell start with # and goes until end of line*
- *Assign a variable*
  - *variable=value*
- *Accessing a variable*
  - *$variable*
- *To separate variable from attached text use {}*

```
Num=13
echo “It’s ${Num}th April today”
```

Varibles

- *Must begin with an alphabet or underscore*
- *Can contain (a to z, A to Z, 0 to 9 and _);*
- *Case sensitive*
  - *no and No are different*
- *Do not use ? , * or some punctuation marks in variable names*

Variables

A variable can be made read-only by using "readonly" command e.g Name="Shell"

```
readonly Name
 Name="Bash"
```

will generate an error

- *Readonly variable is equivalent to the constant once its declared readonly.*
- *Unset can be used to convert it back to modifiable variable*

Variable Types

- ***Local Variables*** *- variables that are present in the current instance of a shell. These will not be available to the programs started by the shell, if they are declared on the command prompt.*
- ***Environment Variables*** *- Environment variables are available to its child processes as well.*
- ***Shell Variables*** *- Special variables that are set by the shell and are required for proper functioning of the shell.*

Variables Cont..

- *All the global environment variables (ENV) can be viewed using*
  - $ *printenv*
- *Displays global as well as local environment variables*
  - $ *set*
- *Displays all global environment variables*
  - $ *env*

Variables Cont..

- *To set any environment variable*
  - $ *export Name=value*
  - $ *set Name=value*

Variables Cont..

- *User wide ENVs are set and configured in*
  - *~/.bashrc*
  - *~/.bash_profile*
  - *~/.bash_login*
  - *~/.profile*
- *System wide ENVs can be configured in*
  - */etc/environment*
  - */etc/profile*
  - */etc/profile.d/*
  - */etc/bash.bashrc*

Commonly used environment variables

- *USER - Gives current User's name*
- *PATH –provides the list of search path*
- *PWD – current working directory*
- *HOME – path of home directory*
- *HOSTNAME – gives the name of the host*
- *LANG – language of the editor*
- *EDITOR – name of the default file editor*
- *UID – User ID*
- *SHELL – current shell*

Special Variables

Variable	*Meaning*
$0	*File name of the current script*
$n	*Here n is an integer which corresponds to the position of an argument at command line*
$#	*No of arguments supplied to the script*
$?	*Exit status of the last command executed*
$$	*Process number of the current shell.*
$!	

Comparators

Separator	*Meaning*
-lt	*Less than*
-gt	*Greater than*
-le	*Less than or equal to*
-ge	*Greater than or equal to*
-e or =	*Equal to*
-ne or !=	*Not equal to*

String comparators

Comparator	*Meaning*
=	*Equal to*
!=	*Not equal to*
<	*Sort string in ascending*
>	*Sort string in descending*

Control Structures – if

if …..; then

elseif….;then

…..

else

…..

fi

Conditions to be tested inside if are written in []

Example for if

```
if [ "$SHELL" = "/bin/bash" ]; then
  echo "your login shell is the bash (bourne again shell)"
else
  echo "your login shell is not bash but $SHELL"
fi
```

While loop

```
while [test condition]
do
 …
 ….
done
```

Example while

```
flag=true

while [ "$flag" = true ]
do
  read choice
  echo $choice
  if [ "$choice" = 'Y' ]; then
     flag=true
     echo "continuing"
  else
     flag=false
  fi
done
```

Case statement

case … in

…)   action on the match;;

esac

Example for Case

```
echo "Enter a number between 1 and 10. "
read NUM
case $NUM in
1) echo "one" ;;
2) echo "two" ;;
 3) echo "three" ;;
4) echo "four" ;;
5) echo "five" ;;
6) echo "six" ;;
7) echo "seven" ;;
8) echo "eight" ;;
9) echo "nine" ;;
10) echo "ten" ;;
*) echo "INVALID NUMBER!" ;;
esac
```

For loop

```
for i in {0..4}          or   for i in somerange
  do
…
…
done
```

- *Conditional exit in for loop - break*

Example of for

```
for i in {0..10}
do
  echo $i
done
```

```
for i in {0..10..3}
do
  echo $i
done
```

Example for

```
for ((  i=1; $i<=5;  i++  ))
do
   echo $i
done
```

Functions

function name()

{…

}

- *Calling a function –*

  use only functionname
- *Local variables – declare using “local” keyword*

Example of functions

```
display()
{
   local local_var=100
   global_var=blessen
   echo "local variable is $local_var"
   echo "global variable is $global_var"
}
```

```
echo "===================="
display
echo "=======outside======="
echo "local variable outside function is $local_var"
echo "global variable outside function is $global_var"
```

THANK YOU