Shell script Assignment

1. Sometimes you really don't want some programs being executed in your system (For example a harmful program of the name say "passwordStealer"). Devise some method that scans for a particular process name-"xyz" at definite intervals, and kills the process –"xyz" automatically, if the process with the name is started. (For E.g. Kill 'vi' editor whenever it is opened). A process can be started any time in another terminal or as a background process by some methods. (Hint: try to get PID from process name; Use 'sleep' function in this exercise).

2. Many times we need to do more with text files than the standard options a text editor like "gedit" would supply us with (e.g. removing all complex numbers in a text file). Try doing these with shell scripts.

 a. Replace all words in the file of the form "date" to Current system date.
 b. Capitalize letters of all words.

3. There is this website where you found a lot of pictures in the url, http://172.16.26.5/images/1.jpg all the way up to http://172.16.26.5/images/72.jpg . Using browser to download the 72 files isn't a good idea. Design an automated **general shell script** application to download all the files to a folder. The script must be flexible enough to get the files from any site say e.g. http://172.16.26.5/pictures/1.jpg. (Hint: use "wget" utility). (Note: to view the images in the above given address in a browser please select the 'no proxy' option in Firefox).

4. Renaming a group of files in a folder is often not very flexible. Find ways to do make the following rename tasks easy, using shell scripts.

 a. Prefix a given name to all files in a folder
  eg. Prefix [MyName] to all files in a folder giving e.g. '[MyName] StarWars.mpg'
 a. Suffix a given name to all files in a folder
  eg. Suffix [MyName] to all files in a folder giving eg. 'Star Wars [MyName].mpg'
 b. Change the extension of a given file type in a folder. (Just rename no need to convert formats).
  eg. Change all 'MPG' file extensions to 'AVI'.

5. A digital clock in the middle of the terminal screen is an eye candy. Create a shell script app that displays a digital clock showing time in 1 second intervals centered in the terminal screen.

6. You have seen watches, mobile phones and even calculators that speak time. Now it's your turn to make a speaking clock using shell scripts. (Hint: Use command line text to speech converter e.g. **festival**. You may need to install it in newer versions of Ubuntu). Feel free to add your own creative modifications.
6.b. Using the above text to speech converter mechanism try to read out a full text file using shell scripting.

7.a. If it was one image you would be happy to use GIMP image editor to edit your image file. But now you are given with many images in a folder to ‘Blur’, make a black and white copy, and then to put each of them in an ornamental border and then ‘Paint’ your name at the top left corner of each image. Write a shell script application that converts all the images in a given directory with the above image manipulations. (Note: See “convert “utility). (Feel free to make your own creative modifications).

7.b. BMP and JPEG are two image file formats. BMP images are one space hungry version of an image file while JPEG is very space efficient. We have a whole folder of BMP files and need to free some space. Compressing to JPEG is a good option. Design a shell script app that can convert all BMP files in a folder to JPEG files. (Note. See ‘**convert**’ utility to convert between file formats).

7.c. Many times you need to resize images to fit your mobile phone screen. Write a shell script application that can resize all the images in a particular directory to a custom resolution (size) (e.g. say 128*144). (Note: See **convert** utility).

8. A shell script can be used simplify repetitive tasks. Write a script file with the name “ez.sh” that take command line arguments such that when the following command is given “ez <command>” one of the following operations must be done, where command can be
-c clear the screen
-d show list of files in current working directory
-b starts the bash shell
-e { editor } start this { editor } if installed

9. In a typical system, the system administrator needs to check the system configurations several times. In this assignment, you need to build a utility "admintool".

When the administrator gives following command at the terminal,

“admintool [option]”

It will show some configurations specified by the option. Feel free to add your own options or a totally new user interface that is best suited for the job (Administrators are often less interested in menus or mouse clicks; it’s all about getting things done faster using commands). The standard options can be as follows;

-l	Currently logged user and his login name
-s	Your current shell
-h	Your home directory
-ost	Your operating system type
-p	Your current path setting
-cw	Your current working directory
-nl	Show Currently logged number of users
-o	About your os and version ,release number , kernel version
-sh	Show all available shells
-m	Show mouse settings
-c	Show computer cpu information like processor type, speed etc

-mem	Show memory information
-hdd	Show hard disk information like size of hard-disk, cache memory, model etc
-fs	File system (Mounted)

10. Many times it is needed that a program should work only for specific users of the system (For example the password changer program must only work for the user "administrator"). Write a shell script that takes a file name as argument and creates the file only if the current user name is 'xyz' else an error message should be displayed.

11. When you have multiple terminals open in different windows you can easily run different applications in each terminal window. But this is not the case when you have a single terminal screen and you need to run different applications simultaneously while being able to switch between them (very handy in server terminals without GUI or while using remote shells like SSH). A single terminal screen to multitask is often a head ache. The 'screen' utility is a workaround in this. Device your own creative script app, that exploits this feature. Demo your app's multitasking capabilities in a single secure shell client terminal.

12. "Error: your Hard Disk is corrupted", The One error message you never want to see but must be prepared for. An auto backup shell script application that backs up your files in a given directory, at specific times (each day/week/month) to a specified location needs to be designed in this exercise.

13. It's a normal behavior in a system that a process in the system hangs, and so we need to kill the process. For a normal user it is easier to kill a process using the process name. Design a utility for a system user- that kills a process when the process name is specified. Sometimes it may be the case that the user does not know the actual process name. For example, the process name may be "crond", but the user knows it as "cron". So if it gives the command "kill cron", then the system should prompt for available options. An example is given below;

```
$ kill cron
Available process: 1. crond 2. cronp
Specify the process number (1 or 2) to kill:
```

14. The **tar** command on your system can not back up files that have more than 100 characters in the path name. Can you design some utility, that can find out such files that cannot be backed up using tar command?

15. For some general purpose editing, you need to find out the number of occurrences of the word "encryption" in a file. Note that a single line can contain more than one occurrences of the word. Design a shell script to make this job easier that finds out number of occurrences of a specific word in a file.