Guided Exercises

1. Study how the shells have been started under the column "Shell Started with..." and complete with the required information:

Shell Started with...	**Interactive?**	**Login?**	**Result of `echo $0`**
`sudo ssh user2@machine2`			
`Ctrl` + `Alt` + `F2`			
`su - user2`			
gnome-terminal			
A regular user uses *konsole* to start an instance of *sakura*			
A script named `test.sh` containing the command `echo $0`			

2. Write the `su` and `sudo` commands to launch the specified shell:

   Interactive-login shell as `user2`

   `su`:

   `sudo`:

   Interactive login shell as `root`

   `su`:

   `sudo`:

   Interactive non-login shell as `root`

   `su`:

   `sudo`:

   Interactive non-login shell as `user2`

   `su`:

`sudo`:

3. What startup file gets read when the shell under “Shell Type” is started?

Shell Type	`/etc/profile`	`/etc/bash.bashrc`	`~/.profile`	`~/.bashrc`
Interactive-login shell as `user2`				
Interactive login shell as `root`				
Interactive non-login shell as `root`				
Interactive non-login shell as `user2`				

Explorational Exercises

1. In Bash we can write a simple `Hello world!` function by including the following code in an empty file:

```
function hello() {
     echo "Hello world!"
}
```

   - What should we do next to make the function available to the shell?

   - Once it is available to the current shell, how would you invoke it?

   - To automate things, in what file would you put the function and its invocation so that it gets executed when `user2` opens a terminal from an X Window session? What type of shell is it?

   - In what file would you put the function and its invocation so that it is run when `root` launches a new interactive shell irrespective of whether it is login or not?

2. Have a look at the following basic, `Hello world!` `bash` script:

```
#!/bin/bash

#hello_world: a simple bash script to discuss interaction in scripts.

echo "Hello world!"
```

   - Suppose we make the script executable and run it. Would that be an interactive script? Why?

   - What makes a script interactive?

3. Imagine you have changed the values of some variables in `~/.bashrc` and want those changes to take effect without a reboot. From your home directory, how could you achieve that in two

different ways?

4. John has just started an X Window session on a Linux server. He opens a terminal emulator to carry out some administrative tasks but, surprisingly, the session freezes and he needs to open a text shell.
   - How can he open that `tty` shell?
   - What startup files will get sourced?

5. Linda is a user of a Linux server. She kindly asks the administrator to have a `~/.bash_login` file so she can have the time and date printed on the screen when she logs in. Other users like the idea and follow suit. The administrator has a hard time creating the file for all other users on the server so he decides to add a new policy and have `~/.bash_login` created for all potential new users. How can the administrator accomplish that task?