

**OCCC CS2183 November 13, 2007**  
**Shell Scripting Programming Homework**  
**Due November 27, 2007**

1. (10%) Write a shell script that takes a directory as an argument and reports the names of all of the zero length ordinary files in the directory. Do appropriate error checking.

2. (10%) Modify the script in Question 1 so that it renames all zero length ordinary files inside of the passed in directory. The files should be renamed using the naming scheme of appending the text `_EMPTY` to the original file name. For example, if we determine that `file1.ps` is empty it should be renamed `file1.ps_EMPTY`.

3. (10%) Write a shell script that accepts an **arbitrary number of arguments** and then reports them back to the user (It should simply loop and echo the position of each argument as well as the argument itself). For example, consider the following sample run of the script titled `argcnt.sh`:

```
$ argcnt.sh arg1 arg2 arg3 arg4 arg5 arg6 arg7 arg8 arg9 arg10
Argument #1 = arg1
Argument #2 = arg2
Argument #3 = arg3
Argument #4 = arg4
Argument #5 = arg5
Argument #6 = arg6
Argument #7 = arg7
Argument #8 = arg8
Argument #9 = arg9
Argument #10 = arg10
```

4. (20%) Write a shell script that implements a simple calculator for integer arithmetic. The calculator should prompt the user and receive it's input in following manner:

```
Enter the first operand: 4
Enter an operator: *
Enter the second operand: 3
Answer: 4 * 3 = 12
```

Assume that in this example the user entered 4, \* and 3 in response to the script's prompting. The permissible operators that your calculator should be able to handle are: `+`, `-`, `*`, `/` and `%`. Your calculator should continue running until the user enters the character `x` at any of the prompts.

5. (20%) Write a number guessing program where the program generates a random number then ask you to guess it. The program will reply with either higher, lower, or correct. The program should repeatedly prompt you until you correctly guess it.

6. (30%) Write a script that identifies all of the users currently logged into the system and then reports information on the number and names of the processes that each user is running. For example, if we find that there are two users currently logged in namely, `jay` and `bill`, then your script should report the following information:

```
Number of Users: 2
User Names: jay, bill
*****
User: jay
Number of Processes: 4
Process List: bash, firefox, emacs, gcc
*****
User: bill
Number of Processes: 2
Process List: bash, pine
*****
Total Number of User Processes: 6
```