LAB 8 — UNIX Shell Scripting

Problem A

1. Specification

Create a Bourne shell script file called **`phone`** that will search a file in your current working directory named **`phone_book.txt`** for telephone numbers and names and print out each line that matches the pattern entered.

2. Implementation

- The script should prompt the user for a name as follows:

```
% phone
Enter the name to search: alex
alex johnson    (416) 555-1234        family doctor
Alexander  Smith (905) 555-9876       home renovation contractor
```

- The pattern to be searched for is case-insensitive. That is, the entries displayed include substrings `alex, ALEX, Alex,` etc.

Problem B

1. Specification

As Problem A. Name the script "**`phone2`**".

2. Implementation

As Problem B, except that the name to be searched for is now entered as a command-line argument:

```
% phone2 alex
alex johnson    (416) 555-1234        family doctor
Alexander  Smith (905) 555-9876       home renovation contractor
```

Problem C

To display a file on/as a web page, it must be readable by all: `chmod a+r file_name`

A directory must be executable and readable by all: `chmod a+rx dir_name`

Write a script called `mkpub` (make public) that takes a directory or file name as a command line argument. It then sets the appropriate permission(s) for the directory or the file, and displays a

confirmation message. If the file/directory does not exist then display an error message as in the example shown below.

Following are a few examples:

```
% mkpub Temp
Directory 'Temp' is now made public.

% ls -ld
drwxr-xr-x  2 utn faculty 4096 Nov 20 18:20 Temp/

% mkpub Temp/example.c
File 'Temp/example.c' is now made public.

% ls -l Temp/example.c
-rw-r--r--  1 utn faculty 0 Nov 20 18:19 Temp/example.c

% mkpub ghost.txt
File 'ghost.txt' does not exist.
```

Problem D

As problem C, except that the user may now enter more than one command-line argument. Following is an example. Name the script "**mkpub2**".

```
% mkpub2 Temp Temp/example.c ghost.txt
Directory 'Temp' is now made public.
File 'Temp/example.c' is now made public.
File 'ghost.txt' does not exist.
```

Problem E

Implement a simple calculator using UNIX shell scripting that accepts input in the following format and displays the result of the computation:

`calc [operand_1] [operator] [operand_2]`

The operands `operand_1` and `operand_2` are integers. The operator is one of the following: addition (+), subtraction (-), multiplication (x), division (/) and modulo (%).

Note: For the multiplication operator in the command line arguments, use letter **`'x'`**. If you use the asterisk **`'*'`**, your program will not work properly.

Hint: Use the `expr` utility. The arithmetic operators are the same as those in C or Java. However, the multiplication operator **`'*'`** requires the use of a backslash in the code, e.g., `$int1 \* $int2`.

Following are a few examples:

```
% calc 200 + 100
300
% calc 100 - 450
-350
% calc -50 x 50
-2500
% calc 30 / 7
4
% calc 29 / 12
2
```

Assume that all command line arguments are valid, and no error checking is required.

Common Notes

Complete the headers in the files to be submitted with your student and contact information as follows:

```
#!/bin/sh
# EECS2031 - Lab 8
# Filename: file_name
# Author: Last name, first name
# Email: Your preferred email address
# Login ID: Your EECS login ID
```