EECS 2031 Section A
Software Tools
Lab 1 (100 points), Version 1

Instructor: Ilir Dema
Release Date: Oct 2, 2022

Due: Oct 17, 2022, midnight

All your lab submissions must be compilable/executable on the department machines. It is then crucial that should you choose to work on your own machine, you are responsible for testing your project before submitting it for grading.

Check the Amendments section of this document regularly for changes, fixes, and clarifications.

Ask questions on the course forum on discord.

1 Policies

- Your (submitted or un-submitted) solution to this assignment (which is not revealed to the public) remains the property of the EECS department. Do not distribute or share your code in any public media (e.g., a non-private Github repository) in any way, shape, or form before you get the permission from your instructors.

- You are required to **work on your own for this lab**. No group partners are allowed.
- When you submit your solution, you claim that it is solely your work. Therefore, it is considered as an violation of academic integrity if you copy or share any parts of your code or documentation.
- When assessing your submission, the instructor and TA may examine your doc/code, and suspicious submissions will be reported to the department/faculty if necessary. We do not tolerate academic dishonesty, so please obey this policy strictly.

- You are entirely responsible for making your submission in time.

- You may submit multiple times prior to the deadline: only the last submission before the deadline will be graded.
- Practice submitting your project early even before it is in its final form.
- No excuses will be accepted for failing to submit shortly before the deadline.
- Back up your work periodically, so as to minimize the damage should any sort of computer failures occur. You can use a **private** Github repository for your labs/projects.
- The deadline is strict with no excuses.
- Emailing your solutions to the instruction or TAs will not be acceptable.

Amendments

So far, so good

2 Problem

In this assignment, you will be writing three shell programs. The first program (`myrm.sh`) is designed to emulate a safer `rm` command. The second program (`guess.sh`) emulates a simple number guessing game. The third program (`calculator.sh`) is a tool that emulates a basic calculator.

3 What to submit

When you have completed the assignment, move or copy your four shell scripts (`myrm.sh, guess.sh, calculator.sh`) in a directory (e.g., a1), zip the whole folder, and submit the zip file on eclass. Make sure you name your files exactly as stated (including lower/upper case letters).

3.1 Safe Delete (33.3%)

The UNIX command `rm` does not provide for recovery of deleted files through a mechanism like the recycle bin typically found in modern operating systems. Write a shell script `myrm.sh` that is designed to replace `rm`. It will move the target file(s) specified as parameters to a recycled bin directory instead of really deleting them. Your script should:

- Create a new directory to be used as the recycle bin.
- Print an error message if there are no files specified as parameters, where the correct usage is also explained.
- Print a feedback message `deleting <filename>` anytime a file with name filename is deleted.

To make it easier, your recycled bin folder must be located under a1 folder: `a1/recycle-bin`. Use the unix `mkdir {p` command to create this directory path inside your script (the –p option of mkdir will create all parent directories up to recycle-bin if they don't exist). Here is a sample output from execution of the program:

```
$ ./myrm.sh b c
deleting b
deleting c
$ ./myrm.sh
Error:no target specified
Usage:./myrm <files>
$
```

Example run

3.2 Guess a Random Integer (33.3%)

Write a shell script "guess.sh" that emulates a simple number guessing game. Your script should:

- Define a random integer in the range of 1 to 64 (inclusive) for the user to guess.
- Display a welcoming message to the user.
- Prompt the user to guess a number between 1 to 64 (inclusive).
- Provide a hint to the user each time he/she enters a guess by displaying a message that reads "Too small." or "Too big." accordingly.
- If a correct guess is entered then display the message "You won!" and exit.
- Allow the user to guess a maximum of 6 times. After the 6th attempt display a message "You lost!" and exit.

Hint: `$RANDOM` returns a different random integer at each invocation in range: 0-32767.

Here is a sample output from multiple executions of the program:

$ ./guess.sh Welcome to the number game. Guess a number between 1 and 64 (inclusive). 32 Too small. Try again. 48 Too big. Try again. 40 Too small. Try again. 44 Too small. Try again. 46 You won! $	$ ./guess.sh Welcome to the number game. Guess a number between 1 and 64 (inclusive). 12 Too small. Try again. 17 Too small. Try again. 22 Too small. Try again. 30 Too small. Try again. 40 Too small. Try again. 50 Too small. You lost! $
Example run 1 (User won!)	Example run 2 (User lost!)

3.3 Basic Calculator (33.3%)

Write a shell script `calculator.sh` that performs basic calculator functions such as addition (+), subtraction (-), multiplication (x) and division (/). The program takes as parameters

- An integer value for the left operand.
- A character that represents the type of operation to be performed, identified by one of the characters ('+', '-', 'x', '/') respectively.
- An integer value for the right operand.

and outputs the result of the operation as an integer. If the requested operation would require a division-by-zero, your program should print an error message `Division-by-zero Error!` and not a result.

Here is a sample output from multiple executions of the program:

```
$ ./calculator.sh
Usage - ./calculator.sh  value1  operator value2
where,
value1: numeric value
value2: numeric value
operator: one of +, -, /, x
$ ./calculator.sh  3 + 4
7
$ ./calculator.sh  3 - 4
-1
$ ./calculator.sh  3 x 4
12
$ ./calculator.sh  3 * 4
Usage - ./calculator.sh  value1  operator value2
where,
value1: numeric value
value2: numeric value
operator: one of +, -, /, x
$ ./calculator.sh  3 / 6
0
$ ./calculator.sh  3 / 3
1
$ ./calculator.sh  3 / 0
Division-by-zero Error!
$
```

Example run