Introduction to Software Systems
Assignment 1 - Shell Programming
Submission Due Date: 21 March 2023

March 14, 2023

Important Note:

- The assignment is an individual submission.
- Maximum Marks = 100
- For queries, reach out to the TAs via Moodle.
- Submissions should be made only on the GitHub repository.
- Every .sh file you submit must be inside the 'submission' directory.

- **Please ensure that you follow the submission format mentioned in each question strictly as you'll be given a zero if you don't because the checking would be automated.**

Question 1

Ask the user for their name and birth date (only month and year). Output their exact age in months (include the current month).**(Total: 5 Marks)**

```
$ bash q1.sh
> Enter Name: Hasbulla Magomedov
> Enter DOB: 07 2002
> Hello Hasbulla Magomedov, your age is 260 months.
```

Input should be taken as:

```
Enter Name: name
Enter DOB: MM YYYY
```

Output should be of the format:

```
Hello <name>, your age is <age in months> months.
```

Submission format: One file named 'q1.sh' in the folder 'submission'.

Question 2:

Managing a personal contact book. (Total: 30 Marks)

Write a shell script to manage your personal contact book which will keep record of your contact details. Follow the below instructions to write the shell script.

- Input: contacts.csv is the input file with fields fname (FirstName), lname (LastName), mobile (MobileNumber), office (CompanyName).
- This contacts.csv will be updated after every insert, delete and update.
- **No command other than 'Display All Contacts', or 'Search and Display' should print anything on the terminal.**
- The output for 'Search and Display' and 'Display All Contacts' should be comma-separated words in the same sequence as the format of the csv.
- The commands you are expected to implement are:
  - Insert Contact
  - Edit Contact
  - Display All Contacts
  - Search and Display
  - Delete Contact

The system should run based on flag-based command line arguments.
A flag `-C` would be used to first specify the command, which would be one of ('insert', 'edit', 'display', 'search', 'delete'). The specifications for each command are as follows:

- Insert Contact (`insert`) **(5 Marks)**

  Description:
  To insert a new contact into the book. You will need to take input of the First Name, Last Name, Contact Number and Company Name, and write it to the CSV. Before adding an entry, you will need to make sure that there does not exist anybody else with the given first name in the book already; if they exist, print: `Unable to insert contact!`.

  Flags:
  - `-f`: for entering first name
  - `-l`: for entering last name
  - `-n`: for entering contact number
  - `-o`: for entering company name

  Examples:
  `./q2.sh -C insert -f FirstName -l LastName -n 1234567890 -o IIIT`

- Edit Contact (`edit`) **(10 Marks)**

  Description:
  To edit an existing contact in the book. The target contact is identified using the first name, which is assumed to be a unique identifier for each entry in the book. Zero or more fields of this contact can be updated using the flags given below.

  Flags:

  - `-k`: specify the first name of the contact to be edited (you can assume that all the first names in the book are unique)
  - `-f`: for entering the new first name
  - `-l`: for entering the new last name
  - `-n`: for entering the new contact number
  - `-o`: for entering the new company name

  Examples:
  ```
  ./q2.sh -C edit -k FirstName -f NewFirstName -l NewLastName -n 0987654321 -o BITS
  ./q2.sh -C edit -k FirstName -f NewFirstName
  ./q2.sh -C edit -k FirstName -l NewLastName
  ./q2.sh -C edit -k FirstName
  ```

- Display All Contacts (`display`) **(5 Marks)**

  Description:
  Read the CSV file and display all the contacts in it. By default, display it in the same order as is in the CSV file, but if flags are provided, sort accordingly.

  Flags:

  - `-a`: to sort in ascending order of first name
  - `-d`: to sort in descending order of first name

  Examples:
  ```
  ./q2.sh -C display
  ./q2.sh -C display -a
  ```

- Search and Display (`search`) **(5 Marks)**

  Description:
  Search the contact book by a specified field given its value and return the results. If multiple results are found, print all of them. If no results are found, print: `No results found!`.

  Flags:

  - `-c`: for the column on which it is to be searched (has to be one of first name, last name, contact number or office)

  - `-v`: for the target value of that column

  Examples:
  ```
  ./q2.sh -C search -c fname -v SomeFirstName
  ./q2.sh -C search -c mobile -v 1234567890
  ```

- Delete Contact (`delete`) **(5 Marks)**

  Description:
  Delete an entry from the contact book. The target contact is identified using the first name, which is assumed to be a unique identifier for each entry in the book.

  Flags:

  - `-k`: specify the first name of the contact to be deleted (you can assume that all the first names in the book are unique)

  Examples:
  ```
  ./q2.sh -C delete -k FirstName
  ```

Use the script named **contact.sh** in the repository to generate **contact.csv** with 100 entries.

Submission format: One file named 'q2.sh' in the folder 'submission'.

Question 3:

Consider a file with a set of words 'words.txt'. **(Total: 16M)**
Each of the following questions needs to have answers appended to a file called 'output_3.txt'. Print all words in the file 'words.txt' which satisfy the following conditions.

1. Words that start with 's' and are not followed by 'a' **(2 Marks)**
2. Words that contain the substring 'ra' in them **(2 Marks)**
3. Words that start with 't' and are followed by 'h' **(2 Marks)**
4. words that contain exactly three consonants in a row **(2 Marks)**
5. Words that are palindrome. example 'racecar' **(2 Marks)**
6. Words that contain only unique characters, such as "quartz" **(2 Marks)**
7. Words that contain both the letters "a" and "e", but not the letter "i" **(2 Marks)**
8. Words that start with a vowel and end with a consonant **(2 Marks)**

Submission format: you are supposed to write a separate script for each of the subquestions named 'q3_⟨subtask number⟩.sh' in the submission folder. For example subtask 2(palindrome) will be named 'q3_2.sh'

Question 4:

File Manipulation (Total: 25 Marks)

1. Remove any words that contain the letter 'j' in the input file and write the other words, one per line to the output file. **(5 Marks)**
2. Remove any words that have fewer than three characters in the input file and write the other words, one per line to the output file. **(5 Marks)**
3. Convert all words to lowercase and randomly shuffle them. **(5 Marks)**
4. Group the words by the number of vowels they contain with a blank line between each group to distinguish them (e.g., all words with 1 vowel should be grouped together, all words with 2 vowels should be grouped together, etc.). The words in each group should be alphabetically sorted. **(10 Marks)**

Submission format: You are supposed to write a separate script for each of the subquestions named 'q4_⟨subtask number⟩.sh' in the submission folder. For example subtask 3 (converting to lowercase) will be named 'q4_3.sh'.

Each of these scripts takes in two filenames as command-line arguments: an input file and an output file. The input file contains a list of words, one per line.

Input format:

```
q4_3.sh <input file name> <output file name>
```

Question 5:

Implement a minimal version of the `tree` command with the following specifications. **(Total: 24 Marks)**

1. The script should accept a single argument that takes the path to a folder and prints the contents of this folder recursively (just like the tree command). This path can be either relative or absolute. **(2.5 Marks)**
2. Implement the flag `-A` which, when provided, will print all the files including the hidden files. **(2.5 Marks)**
3. Implement the flag `-D` which will take an argument of an integer and prints the recursive tree only until the depth argument of the integer. Root folder will be depth 0.
   Print: `Invalid depth!` if the integer input is less than 0. **(4 Marks)**
4. Implement the flag `-a` which will output the files in Ascending order alphabetically in the corresponding depth. **(2.5 Marks)**
5. Implement the flag `-d` which will output the files in Descending order alphabetically in the corresponding depth. **(2.5 Marks)**
6. Implement the flag `-s` which sorts the entire output in ascending order of size in respective depth in the respective folder. **(10 Marks)**

Submission format: Submit one file named `q5.sh` inside the submission folder.
Output format: Follow the exact format mentioned below. `.` is the root directory which is depth 0, `folder1, folder2, folder3` are depth 1, `file1..file5` are same depth under `folder1`.

```
.
|-- folder1
|   |-- file1
|   |-- file2
|   |-- file3
|   |-- file4
|   |-- file5
|-- folder2
|   |-- folder4
|   |   |-- file6
|   |-- folder5
|       |-- file7
|-- folder3
    |-- file8
    |-- folder6
        |-- file9
```