Guidelines:
1. Use of the Internet is not allowed. Please close all browsers on your system. If found using any browser, negative marks will be given. Additionally, disciplinary action will be initiated.
2. Prepare a text file “Your_roll_no.txt” and write the commands used for each question in that file.
3. The exam duration is 2hrs (max). If you have completed early, notify a TA.
4. You have to upload the solution file to canvas before 4:15 PM. Solution uploaded after this time will get a negative 5 marks. If there is some issue, notify a TA beforehand to avoid negative marks.
5. All commands should be compatible with the bash shell. In case, you are using a different shell, change shell to bash for this evaluation.

Section1: Shell Programming

1. Write a shell script to take a number as user input using the command-line argument only and then check whether the number is a palindrome. Note that if the input is not provided as a command-line argument, it should throw an error and should suggest how to use the script. [5]

2. Write a shell script to validate password strength. The conditions to check the strength of the password are as follows.

a. Length: minimum of 8 characters.

b. Contain both alphabet and number.

c. Include both the small and capital case letters.

Missing any of the above conditions will result in a “weaker password” warning and will prompt for a new valid password. The program will only terminate on a successful password entry from the user. [5]

3. Write a shell script to generate a Fibonacci series up to a given range. The range should be user input. [5]

4. Write a shell script to print the following pattern. The line number should be user input. For line number = 5, the pattern should be following. [5]

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

Section 2: Python programming

Write a program that contains the following functions,

1. *read_and_parse(filename)* : Reads the text file, passed as an argument, **line by line**. Breaks line into words. **Strips** whitespaces and punctuation from words, finally converts the word into **lowercase** and **return** list of words. (Note: returned list must not contain string with length zero)
   Hint: import constants named *whitespace* and *punctuation* from *string* module. **[4]**
2. *statistical_analysis(word_list)* : Reads the word list generated in the previous part, creates a dictionary, **key** : word **value** : count (number of times, word has appeared), and **dump** the dictionary as *word_count.pickle*. Also, use this dictionary to find,
   a. **Number of unique words** in the word list.
   b. 5 **most frequent** words.
   c. 5 **least frequent** words.
   d. Words with **mean** frequency.
   e. Words with **median** frequency. **[6]**
3. *sort_and_dump()* : reads the *word_count.pickle,* **sort** the dictionary based on following and write it on two different files.
   a. Sort **alphabetically**.
   b. Sort **based on the frequency**. **[5]**
4. *anagram_list(word_list)* : reads the word list generated in the first part, identify the set of words that are **anagrams** and prints the **five largest set**.
   Hint: Refer problem 9 from lab practice for the definition of anagram words. **[5]**