Institute for Advancing Intelligence (IAI), TCG-CREST
End-Semesteral Examination
Ph.D Program Session: 2024–2025, Semester-I
Subject: Introduction to Programming and Data Structures

Date: 06. 01. 2025 | Maximum Marks: 30 | Time : 3 Hours

Instructions: Answer as much as you can. Use codes from your assignments wherever possible.

. . . . . .

1. Chaining is a common technique used in hash tables to handle collisions. In this method, each slot of the hash table typically contains a linked list to store all the keys that hash to the same index. However, if the distribution function is not chosen wisely, the worst-case search complexity may become $O(n)$, where n is the number of key-value pairs in the table.

   If, instead, a balanced binary search tree (BST) (e.g., an AVL tree) is used in place of a linked list, the worst-case complexity becomes $O(\log(n))$, with a higher probability of better performance compared to using a linked list.

   Problem: Modify your assignment code to use any BST for search, insertion, and deletion operations, replacing the linked list in the chaining mechanism.
   I/O: To test your code, keep I/O same as in the assignment or make your own. [10]

2. Let $G = (V, E)$ be a co-author graph, where V represents the set of nodes (authors) and E represents the edges (co-authorship relations between nodes). Let $F(u)$ denote the set of co-authors of u, i.e., $F(u) = \{v \in V \mid (u, v) \in E\}$. A co-author of a co-author (CoC) of u is any node $w \in V$ such that $w \in F(v)$ for some $v \in F(u)$, but $w \notin F(u)$ and $w \neq u$.

   Problem: Write a program in C to calculate the number of such nodes w, i.e., the number of CoCs of u who are not already a co-author of u.

   Mathematically, given u, find the cardinality of the set:

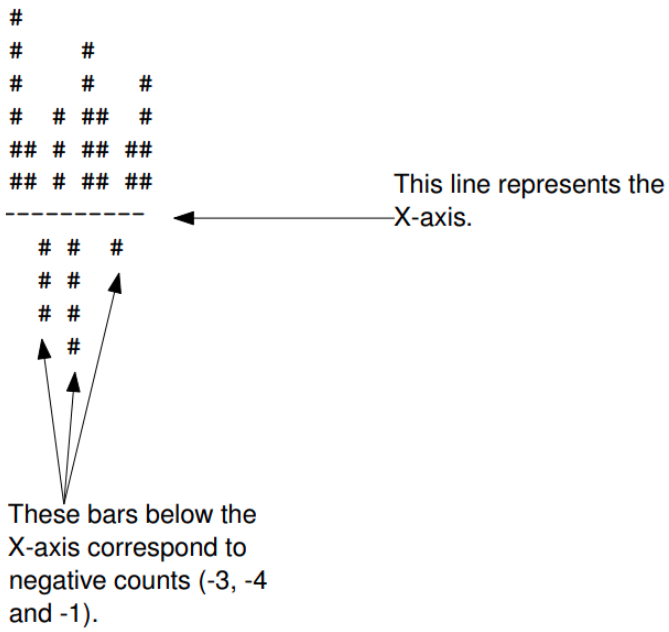
   $$S(u) = \{w \in V \mid w \in F(v) \text{ for some } v \in F(u), w \notin F(u), w \neq u\}.$$

   Input: Graph G in adjacency list form in the file "`input_graph.txt`". u as user input
   Output: $|S(u)|$ [10]

3. Write a program that takes 10 space-separated integers as input and "draws" the corresponding bar diagram on `stdout` or `terminal` or in a file, as shown in the examples below:

   - Each bar should be drawn as a column made up using the `#` character.
   - The height of each bar (i.e., the number of successive lines occupied by the bar) should be equal to the count stored in the corresponding bin.
   - The bars should occupy adjacent columns.
   - If the input contains negative counts, your diagram should contain a line made up using the `-` character to represent the X-axis. Bars corresponding to negative counts should be drawn below the X-axis.

   Example Input: 6 2 -3 3 -4 5 3 -1 2 4
   The output should look like:



[10]

4. Alice wants to send a file to bob in encrypted form. The file contains a paragraph written with capital letters and spaces only. To encrypt the file, `plaintext.txt`, Alice does the following.

   Key generation

   (a) Alice fixes $2 < k < 28$ and generates a $k \times k$ invertable matrices M & computes it inverse M^{-1}.

   (b) Alice share k and M^{-1} with Bob and Keeps k and M

   (c) Each element of M is a element from $\mathbb{Z}_{27}$

   Encryption: Alice wants to encrypt `plaintext.txt` which contains a message with a length that is a multiple of k (Maximum length $100000 * k$).

   (a) Alice reads k letters at a time and maps them into a column vector, denoted as P. The mapping is as follows: space is represented by 0, 'A' is 1, 'B' is 2, $\cdots$, 'Z' is 26.

   (b) She multiplies M and P to obtain the cipher vector C, which is a column vector.

   (c) She converts C back to letters and stores the resulting text in the file `ciphertext.txt`.

   Decryption: Bob has M^{-1}, k and `ciphertext.txt`.

   (a) Bob reads `ciphertext.txt`. He reads k letters at a time and maps them into a column vector Q similar as before.

   (b) He multiplies M^{-1} and Q to obtain the plaintext vector P, which is a column vector.

   (c) He converts P back to letters and stores the resulting text in the file `decryptedtext.txt`.

   Problem: Write a C program to help alice and bob to communicate.

   Input: k and `plaintext.txt`
   Output: M in `secret_alice.txt`, M^{-1} `secret_bob.txt`, `ciphertext.txt`, `decryptedtext.txt`.

   [20]

Appendix: Question 4- Example

Message Representation

Let the message be a sequence of characters consisting of capital letters and spaces only. Each character is represented by a numerical value as follows:

$$\text{Space} = 0, \quad A = 1, \quad B = 2, \quad \ldots, \quad Z = 26.$$

For a given message, the characters are grouped into blocks of size 3. Each block is represented by a column vector $\mathbf{P}$ of size 3×1:

$$\mathbf{P} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix},$$

where p_1, p_2, p_3 are the numerical equivalents of the characters in the block.

Encryption Process

Let Alice's encryption matrix be denoted by M, a random 3×3 matrix. The ciphertext $\mathbf{C}$ corresponding to the plaintext $\mathbf{P}$ is given by the matrix multiplication:

$$\mathbf{C} = M \cdot \mathbf{P}.$$

Each element of the resulting ciphertext vector $\mathbf{C}$ is then converted back to a character using the same mapping from numbers to characters:

$C_1 \rightarrow$ character corresponding to C_1, $\quad C_2 \rightarrow$ character corresponding to C_2, $\quad C_3 \rightarrow$ character corresponding to

Decryption Process

Bob receives the ciphertext vector $\mathbf{C}$, and to recover the original plaintext, he multiplies $\mathbf{C}$ by the inverse of the encryption matrix M^{-1}:

$$\mathbf{P} = M^{-1} \cdot \mathbf{C}.$$

Bob then converts the resulting numbers back to characters to obtain the decrypted message.

Example

Let us take the encryption matrix M as:

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

Suppose Alice wants to encrypt the block "TCG". The corresponding vector is:

$$\mathbf{P} = \begin{pmatrix} 20 \\ 3 \\ 7 \end{pmatrix}.$$

The ciphertext vector is:

$$\mathbf{C} = M \cdot \mathbf{P} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \cdot \begin{pmatrix} 20 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 1 \times 20 + 2 \times 3 + 3 \times 7 \\ 4 \times 20 + 5 \times 3 + 6 \times 7 \\ 7 \times 20 + 8 \times 3 + 9 \times 7 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 20 + 6 + 21 \\ 80 + 15 + 42 \\ 140 + 24 + 63 \end{pmatrix} = \begin{pmatrix} 47 \\ 137 \\ 227 \end{pmatrix}.$$

Now, to convert this back to characters, we take each element modulo 27 (since there are 27 possible characters: 26 letters and the space):

$$C_1 = 47 \mod 27 = 20 \quad (\text{corresponding to } T),$$
$$C_2 = 137 \mod 27 = 2 \quad (\text{corresponding to } B),$$
$$C_3 = 227 \mod 27 = 8 \quad (\text{corresponding to } H).$$

Thus, the encrypted message corresponding to the block "TCG" is "TBH".