CIS 22C - Data Structures
Team Project Variations

Project Option #1: BST, Hash Table, and Self-Adjusting Linked Lists

1. Solve collisions using **linked list** resolution.
2. A self-adjusting list is like a regular list, except that all insertions are performed at the front, and when an element is accessed by a search, it is moved to the front of the list, without changing the relative order of the other items. The elements with highest access probability are expected to be close to the front.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Longest Linked List
    c. Average number of nodes in linked lists

Project Option #2: BST, Hash Table – Quadratic Probe, and Stacks

1. The user can undo the delete in the reverse order of the delete sequence. When the user selects "Save to file", the undo **stack** is cleaned out (no undo possible unless more delete occurs first). The head node will contain one more pointer: to the stack header, and one more option needs to be added to the menu: "Undo delete".
2. Collision resolution method: open addressing – **quadratic probe**.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Number of Collisions
    c. Longest Collision Path

Project Option #3: BST, Hash Table – Bucket Hashing, and Queues

1. One approach to handling collisions is to hash to buckets. A bucket is a structure that accommodates multiple data occurrences **(Bucket Hashing)** In this case, the head node, listHead, will contain one more field: bucket size (the number of records stored in a "bucket") and arySize represents the number of buckets. If you have to insert into a full bucket, print a message (insert not possible at this time). Set bucket size to 3.
2. Save data to file using the breadth-first traversal of the BST (**queue**)
3. Efficiency:
    a. Load Factor
    b. Longest bucket
    c. Number of Collisions

Project Option #4: BST, Hash Table, and Linked Lists

1. Solve collisions using **linked list** resolution.
2. The linked list data must be sorted.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Longest Linked List
    c. Average number of nodes in linked lists 8

Project Option #5: BST, Hash Table – Variation of Linear Probe, and Stacks

1. The user can undo the delete in the reverse order of the delete sequence. When the user selects "Save to file", the undo **stack** is cleaned out (no undo possible unless more delete occurs first). The head node will contain one more pointer: to the stack header, and one more option needs to be added to the menu: "Undo delete".
2. Collision resolution method: open addressing – **variation of linear probe**: when inserting, add 1 subtract 2, add 3, subtract 4, until we locate an empty element.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Number of Collisions
    c. Longest Collision Path 9

Project Option #6: BST, Hash Table – Pseudorandom, and Stacks

1. The user can undo the delete in the reverse order of the delete sequence. When the user selects "Save to file", the undo **stack** is cleaned out (no undo possible unless more delete occurs first). The head node will contain one more pointer: to the stack header, and one more option needs to be added to the menu: "Undo delete".
2. Collision resolution method: open addressing – **pseudorandom**.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Number of Collisions
    c. Longest Collision Path

Project Option #7: BST, Hash Table – Key Offset, and Stacks

1. The user can undo the delete in the reverse order of the delete sequence. When the user selects "Save to file", the undo **stack** is cleaned out (no undo possible unless more delete occurs first). The head node will contain one more pointer: to the stack header, and one more option needs to be added to the menu: "Undo delete".
2. Collision resolution method: open addressing – **key offset**.
3. Save data to file in hash table sequence.
4. Efficiency:
    a. Load Factor
    b. Number of Collisions
    c. Longest Collision Path