Thien An Nguyen, NetID: kz7962
Timothy Miu, NetID: fb5679
Ghazi Noseir, NetID: fr3474

Project Name: Unsorted List Implementation and Testing

Objective: The objective of this project is to implement and test an unsorted list data structure using C++. The project involves creating the necessary classes and functions to support common list operations and validating the correctness of these operations through a driver program.

Class Names:

- UnsortedType: Represents the unsorted list data structure.
- ItemType: Represents individual elements in the list.

Functions to Implement:

- In the UnsortedType class:
  - UnsortedType(): Constructor to initialize the list.
  - void MakeEmpty(): Function to clear the list.
  - bool IsFull() const: Function to check if the list is full.
  - int GetLength() const: Function to get the number of elements in the list.
  - ItemType GetItem(ItemType, bool&): Function to retrieve an item from the list.
  - void PutItem(ItemType item): Function to add an item to the list.
  - void DeleteItem(ItemType item): Function to remove an item from the list.
  - void ResetList(): Function to reset the current position for iteration.
  - ItemType GetNextItem(): Function to get the next item during iteration.
- In the ItemType class:
  - ItemType(): Constructor to initialize an item.
  - RelationType ComparedTo(ItemType) const: Function to compare items.
  - void Print(std::ostream&) const: Function to print an item.
  - void Initialize(int number): Function to initialize an item's value.

Test Cases:

- Testing Initialization:
  - Initialize an empty list and check its length.
  - Initialize an item and validate its value.
- Testing PutItem and GetItem:
  - Add items to the list using PutItem and verify that they are present in the list using GetItem.
- Testing DeleteItem:
  - Add items to the list, delete them using DeleteItem, and confirm their removal.

Thien An Nguyen, NetID: kz7962
Timothy Miu, NetID: fb5679
Ghazi Noseir, NetID: fr3474

- Testing GetLength and IsFull:
  - Add items to the list and check the length.
  - Add items until the list is full and verify that IsFull returns true.
- Testing ResetList and GetNextItem:
  - Add items to the list, reset the list's position, and iterate through the list using GetNextItem. Validate the order.
- Testing MakeEmpty:
  - Add items to the list and then clear the list using MakeEmpty. Check if the list is empty.
- Testing Invalid Commands:
  - Include some invalid commands in the input data and ensure that the program handles them gracefully, printing an error message.
- Testing Input File Execution:
  - Create a test input file with a sequence of commands similar to the provided example (listdata).
  - Run the program using this input file and compare the output with the expected results.

Expectations:

- The UnsortedType class should correctly implement all list operations.
- The ItemType class should support item comparison and printing.
- The driver program (listDriver.cpp) should read commands from an input file, execute them, and produce output in the specified format.
- The program should handle invalid commands gracefully and not crash.

Description of Logic: The logic of this project involves implementing the core functionality of an unsorted list.

- The UnsortedType class maintains an array of items and supports operations like insertion, deletion, retrieval, iteration, and more.
- The ItemType class defines individual list items with comparison and printing capabilities.
- The driver program (listDriver.cpp) acts as the user interface, allowing users to provide commands via an input file. It reads these commands, executes the corresponding operations on the list, and produces output based on the results.