Name: Alexis Ouellette
Date: 07/25/2023
Assignment: Program 2

Pseudo-code of Algorithm:

1. Classes: Bank to represent a list of items in a Bank object.
2. Main function:
3. Define Bank objects: south and north, string farmer location, string user choice of item, char user choice to play agin, bool validate move, out stream file variable to hold name of file
4. Name file name
5. Display game instructions
6. Initialize the game state: south = fox, chicken, grain; north = empty; farmer's location = south bank
7. Do-while (user wants to play)
8. while(checkGameState(Bank, Bank, string = true)
9. Display South bank, north bank, and farmers location
10. Save to displays to file
11. Get user input for item choice
12. Print the farmer took it with him
13. Remove and add element to opposite banks.
14. Move the farmer to the opposite bank.
15. Display update banks
16. Save to file
17. Check game condition
18. If game is lost: print explanation, and ask user to play again.
19. Initialize game state and restart game is yes, end if no
20. Repeat is game is won.

Class(es):

Bank class: to hold the items on each bank

- Private members:
  - Structure BankNode: represents a single item of a list. I used a structure to allow easy access to the nodes.
  - BankNode pointer to the head of the node structure and to the tail of the node structure
- Public members:
  - Bank(): constructor to initialize the head and tail of the node to null.
  - ~Bank(): destructor to free the memory.
  - Int getLength(): returns the number of nodes in a list object, used to calculate win in the main function.
  - Void addItem(string): allows for the insertion of a node to an empty list or an occupied list.
  - Void removeItem(string): allows for the removal of a node from a list object.
  - Void displayBank(): displays the currents items that are in a list object.
  - Bool containsItem(string): searches through the list to see if the item is there.
  - Void deleteAll(): deletes the entire list to allow for re-initialization and initialization of the current game state.

Summary:

Design Decisions

I used a singly-linked list to hold the items of the bank. Since the game had a a simple logic to it, the linked list worked great. Easy for insertion, removal, searching, etc., for a program of this capacity.

Issues

The only issued I had was making it too complicated.

Notes

Anything interesting you want to share about your design or anything else about this programming assignment?