PROBLEM: WORD LIST SORTER AND REVERSER (J. Santillana)

The Word List Sorter and Reverser will allow the user to come up with a word list of up to a number of items of their choosing (maximum is 1000). Users are not allowed to add duplicate words into the list. The user may also choose to terminate the input process before reaching the set number of items. Once inputting is done, the list is then sorted in a desired order(ie. Either ascending or descending). The list will be sorted based on the user's choice(Ascending or Descending) afterwhich the words in the list will then be scrambled in the reverse order(eg. Exam will be displayed as maxE) and then displayed.

The following aliases will be used for this program:

Boolean - an integer type to be used for true or false values
String20 - a character array that will be used for strings that have a max of 20 characters

It will also have the following constants:

EXIT - it has an assigned value of **("0")**. This will act as the sentinel value for the input process.
FALSE - it has an assigned value of **(0)**. To be used with the Boolean alias.
TRUE - it has an assigned value of **(!FALSE)**. To be used with the Boolean alias.

Your task is to edit and complete the skeleton file, **LASTNAME-WORDLIST.c**, and implement the 5 user-defined functions as described below.

1. **`int inputWords(String20 words[], int size)`** - this function will be used for getting the word inputs from the user. It will check if the inputted word is already in the list. If it does, it will ask the user to enter a different word. A duplicate word should not be counted in the total count of items. The input process ends if either the user inputs up to the number of selected items or if the user inputs 0. It will count the number of items inputted into the list and return the item count.

   ** test inputWords() **
   test 1: Assume 5 was the inputted number of list items

   magic
   test
   certain
   apple
   news

   Output:
   The words inputted:
   magic
   test

certain
apple
news

test 2: Assume 4 was the inputted number of list items but the user decided
to only input 2 items

quiz
notes
0

Output:
The words inputted:
quiz
notes

test 3(input duplicate words): Assuming 2 was the inputted number of list items

pencil
pencil

Output:
The words inputted:
pencil
pencil
Matching word found! Please input a different word!

2. **`int checkDuplicate(String20 words[], String20 word, int size)`** - this function will check if the word already exists in the list and return TRUE if the string already exists otherwise it will return FALSE.

3. **`void reverseWords(String20 words[], int size)`** - this function will reverse the words on the list. (Eg. Exam will become maxE)

** test reverseWords() **
test 1: Assume 4 was the inputted number of list items

laptop
mouse
scanner
tablet

Output:

After Reversing:
potpal
esuom
rennacs
telbat

4. **`void sortItems(String20 words[], int size, int sortOrder)`** - this function will sort the list according to the selected sorting order chosen by the user.

   ** test sortItems() **
   test 1: Assume 3 was the inputted number of list items

   dragon
   wizard
   knight

   Output: Assuming Ascending was selected
   Words in Ascending order:
   dragon
   knight
   wizard

   test 2: Assume 5 was the inputted number of list items

   orange
   blueberry
   apple
   strawberry
   grape

   Output: Assuming Descending was selected
   Words in Descending order:
   strawberry
   orange
   grape
   blueberry
   apple

5. **`void rotateLeft(String20 words[], int size, int m)`** - this function will shift the words to the left by m slots where m >= 0.
   For example: Let set1[] = { "apple", "banana", "chico", "durian" };
   Let set2[] = { "ant", "bear", "cat", "dog", "elephant" };

* Example 1: rotateLeft(set1, 4, 0); then set1 will remain the same. i.e., no shifting applied.
* Example 2: rotateLeft(set1, 4, 2); then set1 will become { "chico", "durian", "apple", "banana" };
* Example 3: rotateLeft(set2, 5, 5); then set2 will remain the same
* Example 4: rotateLeft(set2, 5, 6); then set2 will become { "bear", "cat", "dog", "elephant", "ant" };

You are given the following files for this problem:

- **`LASTNAME-WORDLIST.c`** - The file you will be working with where you will be placing your solution.
- **`main.c`** - file that contains the main function. You will test your solution by compiling this file.
- **`defs.h`** - header file with constant definitions, typedefs and function prototypes. This is needed by the main.c file when testing your solution.
- **`INPUT1.txt`** - contains example input data. Test your solution by creating your own list with your own data.
- **`EXPECTED1.txt`** - contains the expected results when the exe file of your solution is executed using data stored in the INPUT1.txt file.
- **`INPUT2.txt`** - contains example input data, specifically testing for duplicate entries and the “0” sentinel value. Test your solution by creating your own list with your own data.
- **`EXPECTED2.txt`** - contains the expected results when the exe file of your solution is executed using data stored in the INPUT2.txt file.

TESTING & SCORING:

- Your program will be tested using a different input text file (containing different words).
- Each correct function definition will be given **10 points** each. Thus, the maximum total score will be 50/50.
- **A function that has a syntax/compilation error will result in a score of 0.**
- **A function that does not produce any of the required output (for example, due to runtime error) will result in a score of 0.**

`----------NOTHING FOLLOWS----------`