Assignment 12: Delete Repeats

Due: December 9, 2019 (Thursday)

All programs must be correctly run for them to be considered as accepted submissions. (Per each missing- or unaccepted- submission, your grade will be lowered by one letter grade); The last day to upload the assignments is December 17.

Objective

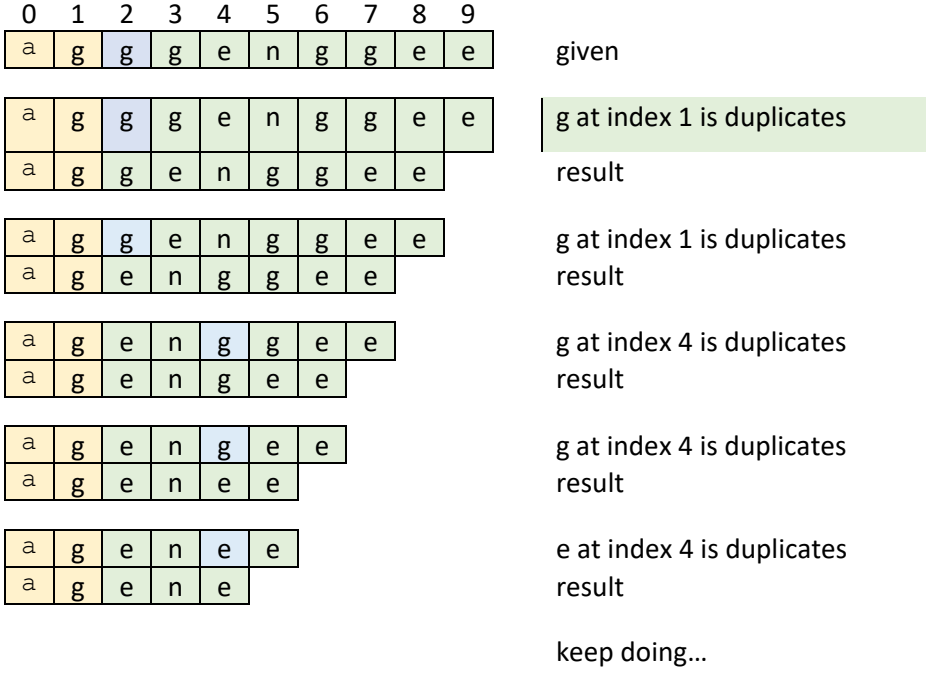
This assignment is to introduce arrays, partially filled arrays and arrays with functions.

Code : deleteRepeats .cpp
Input : none
Output : none

Assignment: Delete Repeats

Write a program that deletes repeated entries in a character array. Please note that there is not delete feature in C++. You are going to implement it. The array size and the array entries are user inputs. Once one set of characters are processed, the program should ask if the user wants to repeat for another round and repeat the same until user enters N or n (representing no).
For that, write and use at least 3 user defined functions given later in this assignment. (You **must use local variables; no global variables; you must NOT use c-style strings; use an array of characters.** Use pass-by-reference and pass-by-value appropriately):

The definition of delete:
If you find duplicate character, all the characters but the first appearance should be deleted, meaning, you manually need to move the rest of the characters left such that the next immediate character will replace the duplicate one (except this rearrangement, the characters will not actually be deleted even though I called that way). The last character in the array will still be there as a result of the shift. We do not care about it because we only care the active characters in the array.
The following example only shows the active characters.
e.g.,



1. **ReadArray**: this function reads both the array size and array elements from user.

2. **DeleteRepeats** : This function takes a partially filled array of characters as a formal parameter and deletes all repeated characters.
   Since a partially filled array requires two arguments, the function will actually have two formal parameters: an array parameter and a formal parameter of integer type that gives the number of array positions used.

   When a letter is deleted, the remaining letters are moved forward to fill in the gap. This will create empty positions (at the end of the array positions that we do not need anymore) so that less of the array is used. The parameter that tells the number of positions used in the array, as well as the array itself, will then be updated accordingly.
   Print the updated array after each character deletion. (see the sample output)

3. **PrintArray**: takes two parameters (the array and the used array size)

Sample Output

Note that the positions are referred in terms of the position of the number; not the index.

```
what is the size 10
Enter the array
a g g g e n g g e e

found duplicates at 2 and 3 : g and g
Updated array: a g g e n g g e e

found  duplicates at 2 and 3 : g and g
Updated array: a g e n g g e e

found duplicates at 2 and 5 : g and g
Updated array: a g e n g e e

found duplicates at 2 and 5 : g and g
Updated array: a g e n e e

found duplicates at 3 and 5 : e and e
Updated array: a g e n e

found duplicates 3 and 5 : e and e
Updated array: a g e n

The array after delete repeats
a g e n
Repeat? Y

what is the size 10
Enter the array
a g e n g g e e h h

found duplicates at 2 and 5 : g and g
```

```
Updated array: a g e n g e e h h

found duplicates  at 2 and 5 : g and g
Updated array: a g e n e e h h

found duplicates  at 3 and 5 : e and e
Updated array: a g e n e h h

found duplicates  at 3 and 5 : e and e
Updated array: a g e n h h

found duplicates  at 5 and 6 : h and h
Updated array: a g e n h

The array after delete repeats
a g e n h
Repeat? N
```