

I tried at first using a hashmap, but we asked Dr. Asish if you actually need one since this specific problem can be made w/  $O(n)$  complexity (like a hashmap), and he said it's okay to submit without using hashmap.

## CODE:

```
// Given an array of words as input. Display the words and their corresponding
length. Also
/*display the biggest word.
    Input array:{"apple", "banana", "Floridapoly","Algorithm", "data"};
    Output:
    apple 5
    banana 6
    Floridapoly 11
    Algorithm 9
    data 4
    Biggest word: Floridapoly
-----
// Don't submit c++ code. Submit a screenshot that includes code and output. */
#include <iostream>
#include <string>
#include <unordered_map>
#include <stack>
using namespace std;

int main() {
    // Input array of words
    string words[] = {"apple", "banana", "Floridapoly","Algorithm", "data"};

    int sizeArr = sizeof(words)/sizeof(words[0]);
    //variable for largest size of the string
    int largestStringLength = 0;
    int largestIndex = 0;

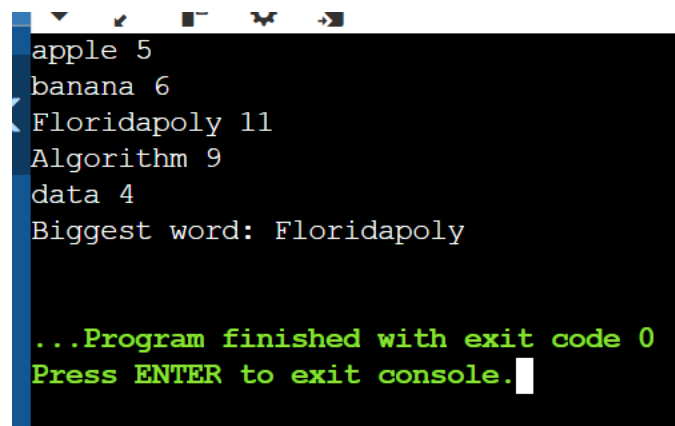
    //iterate through the word array
    for(int i = 0; i<sizeArr; i++) {
        //make size variable for length of current string
```

```

        int size = words[i].size();
        //compare size variable to largest string length, if greater than, update
largest string length & index accordingly
        if (size > largestStringLength) {
            largestStringLength = size;
            largestIndex = i;
        }
        //print word and the size
        cout << words[i] << ' ' << size << endl;
    }
    //print the biggest word using the largest index
    cout << "Biggest word: " << words[largestIndex] << endl;
    return 0;
}

```

### Output:



```

apple 5
banana 6
Floridapoly 11
Algorithm 9
data 4
Biggest word: Floridapoly

...Program finished with exit code 0
Press ENTER to exit console.

```