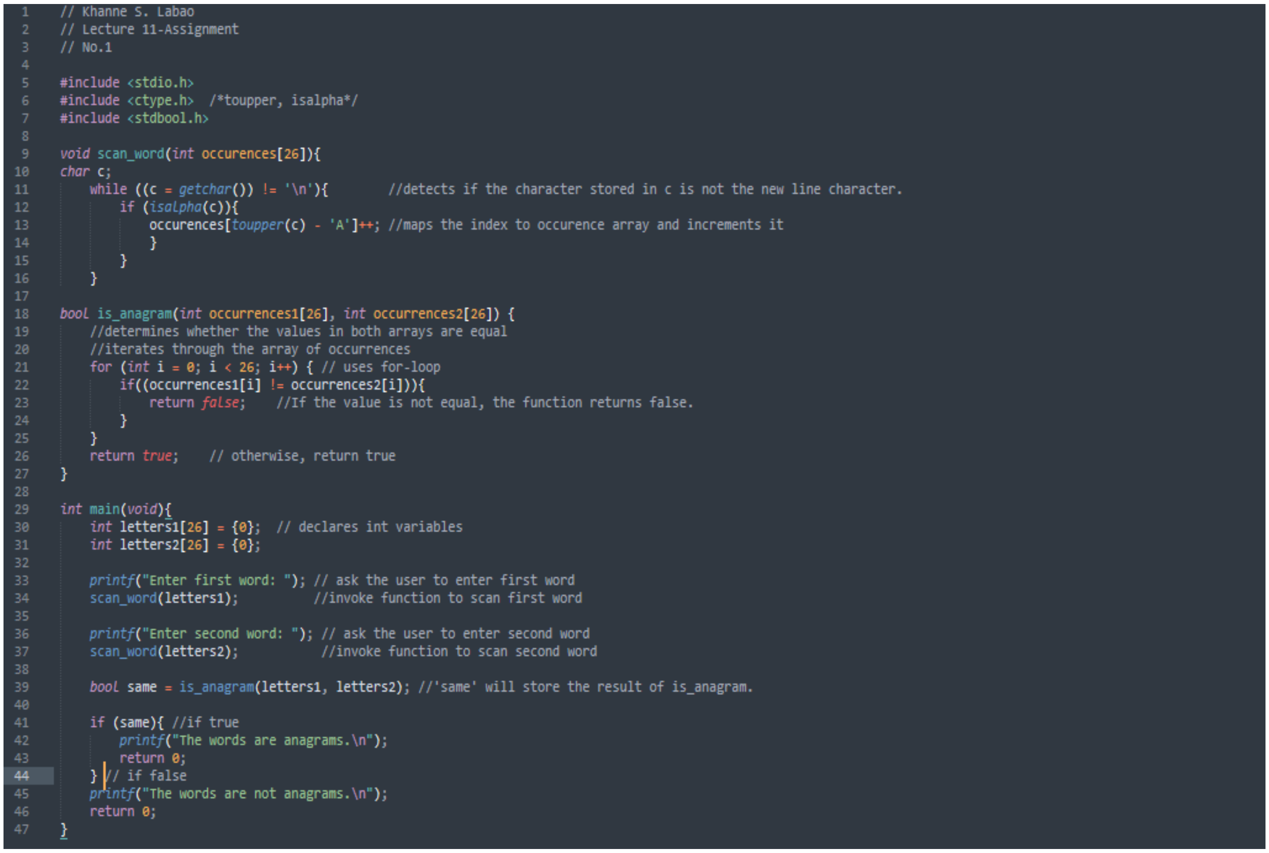
CMSC 21 Lec11
Pointers and Multidimensional Arrays
Assignment

Name: Khanne S. Labao
Student number: 202009642
Course and Year: BS in Computer Science 1

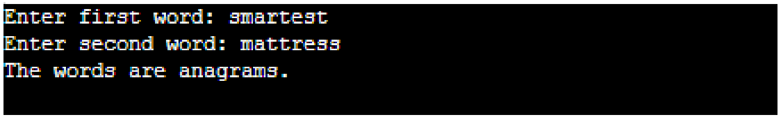
Github link: https://github.com/Khanne02/CMSC-21-Lec11--Assignment

- Assignment no.1

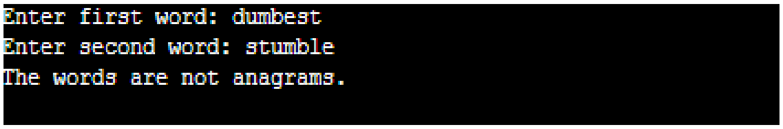


```c
// Khanne S. Labao
// Lecture 11-Assignment
// No.1

#include <stdio.h>
#include <ctype.h>  /*toupper, isalpha*/
#include <stdbool.h>

void scan_word(int occurences[26]){
char c;
    while ((c = getchar()) != '\n'){        //detects if the character stored in c is not the new line character.
        if (isalpha(c)){
            occurences[toupper(c) - 'A']++; //maps the index to occurence array and increments it
            }
        }
    }

bool is_anagram(int occurrences1[26], int occurrences2[26]) {
    //determines whether the values in both arrays are equal
    //iterates through the array of occurrences
    for (int i = 0; i < 26; i++) { // uses for-loop
        if((occurrences1[i] != occurrences2[i])){
            return false;    //If the value is not equal, the function returns false.
        }
    }
    return true;    // otherwise, return true
}

int main(void){
    int letters1[26] = {0};  // declares int variables
    int letters2[26] = {0};

    printf("Enter first word: "); // ask the user to enter first word
    scan_word(letters1);          //invoke function to scan first word

    printf("Enter second word: "); // ask the user to enter second word
    scan_word(letters2);           //invoke function to scan second word

    bool same = is_anagram(letters1, letters2); //'same' will store the result of is_anagram.

    if (same){ //if true
        printf("The words are anagrams.\n");
        return 0;
    } // if false
    printf("The words are not anagrams.\n");
    return 0;
}
```

Output

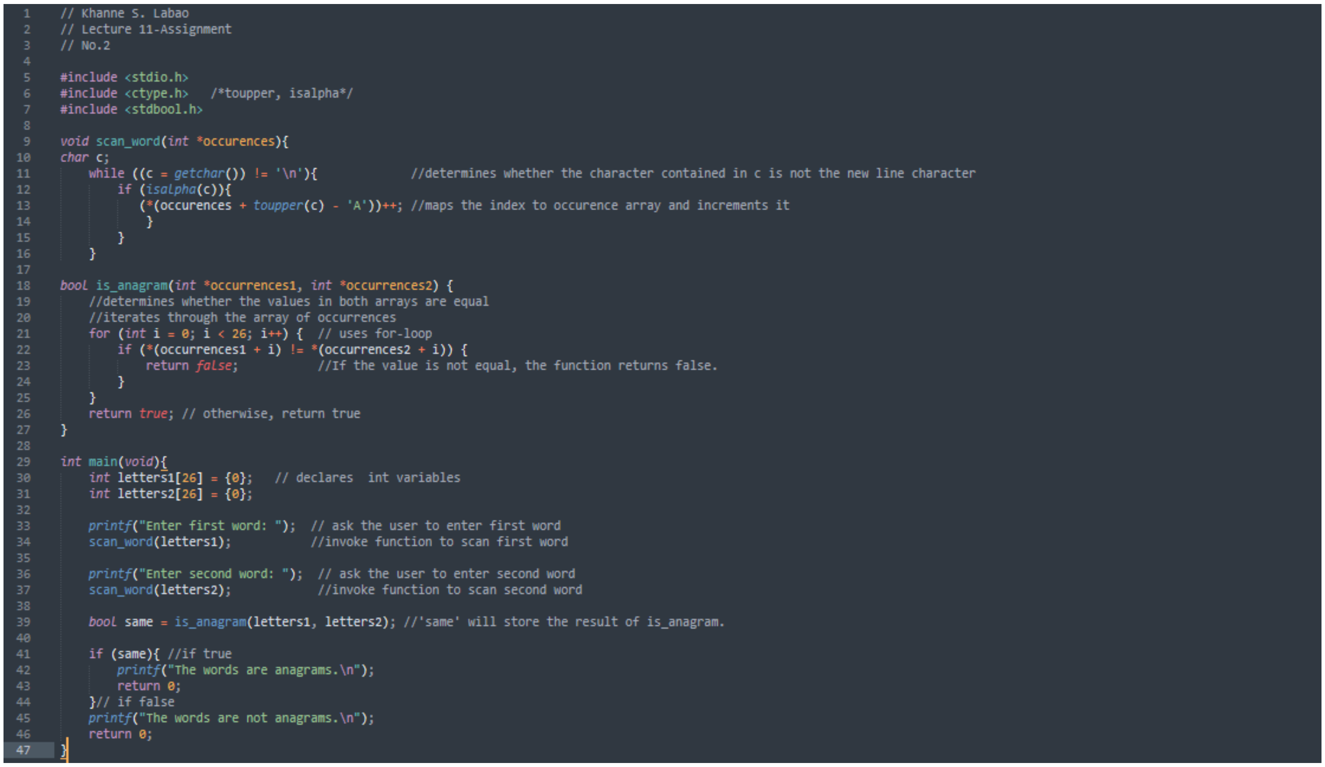


```
Enter first word: smartest
Enter second word: mattress
The words are anagrams.
```



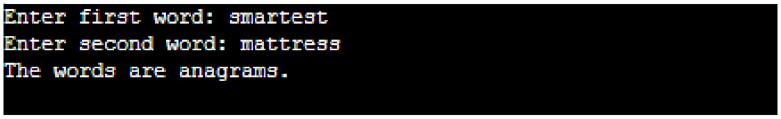
```
Enter first word: dumbest
Enter second word: stumble
The words are not anagrams.
```

- Assignment no. 2

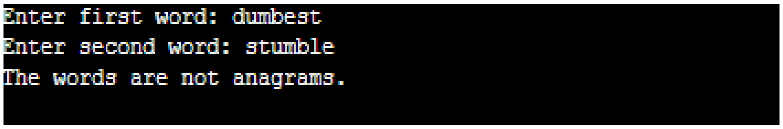


```c
// Khanne S. Labao
// Lecture 11-Assignment
// No.2

#include <stdio.h>
#include <ctype.h>   /*toupper, isalpha*/
#include <stdbool.h>

void scan_word(int *occurences){
char c;
    while ((c = getchar()) != '\n'){             //determines whether the character contained in c is not the new line character
        if (isalpha(c)){
           (*(occurences + toupper(c) - 'A'))++; //maps the index to occurence array and increments it
            }
        }
    }

bool is_anagram(int *occurrences1, int *occurrences2) {
    //determines whether the values in both arrays are equal
    //iterates through the array of occurrences
    for (int i = 0; i < 26; i++) {  // uses for-loop
        if (*(occurrences1 + i) != *(occurrences2 + i)) {
            return false;           //If the value is not equal, the function returns false.
        }
    }
    return true; // otherwise, return true
}

int main(void){
    int letters1[26] = {0};   // declares  int variables
    int letters2[26] = {0};

    printf("Enter first word: ");  // ask the user to enter first word
    scan_word(letters1);           //invoke function to scan first word

    printf("Enter second word: ");  // ask the user to enter second word
    scan_word(letters2);            //invoke function to scan second word

    bool same = is_anagram(letters1, letters2); //'same' will store the result of is_anagram.

    if (same){ //if true
        printf("The words are anagrams.\n");
        return 0;
    }// if false
    printf("The words are not anagrams.\n");
    return 0;
}
```

Output



```
Enter first word: smartest
Enter second word: mattress
The words are anagrams.
```



```
Enter first word: dumbest
Enter second word: stumble
The words are not anagrams.
```

Explanation

- In no. 2, **void scan_word(int occurrences [26])** will be replaced by **void scan_word(int *occurrences)**, and **bool is_anagram(int occurrences1 [26], int occurrences2 [26])** will also be replaced by **bool is_anagram(int *occurrences1, int *occurrences2)** and then passed to the function is_anagram, which will return true if the items in the two words are the same (including that they are anagrams) and false otherwise.