“2942.Find Words Containing Character"

Question :

You are given a 0-indexed array of strings words and a character x.

Return an array of indices representing the words that contain the character x.

Note that the returned array may be in any order.

Constraints :

- 1 ≤ words.length ≤ 50
- 1 ≤ words[i].length ≤ 50
- x is a lowercase English letter.
- words[i] consists only of lowercase English letters.

Inputs :

- words: An array of strings.
- x: A character to search for in the strings.

Outputs :

- An array of indices representing the words containing the character x.

Example 1 :

Input:
words = ["leet", "code"], x = "e"

Output:
[0, 1]

Explanation:
The character "e" occurs in both words: "leet" and "code".

Example 2 :

Input:
words = ["abc", "bcd", "aaaa", "cbc"], x = "a"

Output:
[0, 2]

Explanation:
The character "a" occurs in "abc" and "aaaa".

Example 3 :

Input:
words = ["abc", "bcd", "aaaa", "cbc"], x = "z"

Output:
[]

Explanation:
The character "z" does not occur in any of the words.

Algorithm :

1. Allocate memory for the result array dynamically using malloc.
2. Initialize an index counter to keep track of the current position in the result array.
3. Loop through the array of words.
    - For each word, check if the character x exists in the word using strchr().
    - If the character is found, add the current index to the result array.
    - Resize the array using realloc() to ensure enough space for additional indices.
4. Update the return size to the total number of indices found.
5. Return the array of indices.

Code :

```
#include <stdlib.h>
#include <string.h>

int* findWordsContaining(char** words, int wordsSize, char x, int* returnSize)
{
    int* indexes = (int*)malloc(sizeof(int) * 1);
    int index = 0;

    for (int i = 0; i < wordsSize; i++) {
        if (strchr(words[i], x)) {
            indexes[index] = i;
            indexes = (int*)realloc(indexes, sizeof(int) * (index + 2));
            index++;
        }
    }
    *returnSize = index;
    return indexes;
}
```

Time Complexity :

- O(n * m): Where n is the number of words and mmm is the average length of the words. The function strchr() scans each word, leading to this complexity.

Edge Cases :

1. Character not found in any word:
    - Example: words = ["abc", "def"], x = "z" → Output: [].
2. All words contain the character:
    - Example: words = ["abc", "adc", "aec"], x = "a" → Output: [0, 1, 2].
3. Single word in the input:
    - Example: words = ["abc"], x = "b" → Output: [0].
4. Multiple occurrences of the character in a word:
    - Example: words = ["aaa", "aba"], x = "a" → Output: [0, 1] (no duplicates in output).