Laboratorium 11 - Teksty i napisy

Zadanie 1 - słowa

Dany jest program, który wypisuje wszystkie słowa z podanego zdania i wyświetla ich liczbę.

```
#include <stdio.h>
#include <string.h>
#define SIZE 100

int count_strings(char * text, char words[][SIZE]);
void show_words(char words[][SIZE], int count);

int main() {
    char text[] = "Programowanie w C jest fajne";
    char words[SIZE][SIZE];
    int strings_count =  count_strings(text, words);
    printf("Ilosc slow to: %d\n",strings_count);
    show_words(words, strings_count);

    return 0;
}
```

Uzupełnij funkcje:

```
int count_strings(char * text, char words[][SIZE]);
void show_words(char words[][SIZE], int count);
```

Wynik działania programu:

```
Ilosc slow to: 5
Programowanie
w
C
jest
fajne
```

Zadanie 2 - dwa typy

Napisz program, który pobiera od użytkownika dwie zmienne, jedną w formacie `integer`, drugą w formacie `dobule`. Używając funkcji `sprintf` zapisz pobrane zmienne do jednej tablicy `char`. W tablicy powinna znaleźć się też informacja jakiego typu jest dana zmienna. Wykorzystując tylko jedno wywołanie funkcji `printf` wyświetl zawartość tablicy. Przykładowe działanie programu:

```
Wprowadź zmienną typu integer oraz zmienną typu double: 10 11.2345
Sformatowane dane:
integer:    10
double:  11.23
```

Zadanie 3 - numer wyrazu

Wypełnij prototypy funkcji:

```
... count_first_sequence(char * text, ...);
void print_sequence(...);
void recalculate_sequence(...);
```

Funkcja `count_first_sequence` zamienia wyraz na ciąg liczb naturalnych taki, że poszczególne elementy ciągu oznaczają liczbę wystąpień konkretnej litery w tekście, np. dla wyrazu: `niebieski` ciąg wygląda następująco: `1 3 2 1 1 1` , gdzie:

- 1 ← liczba wystąpień litery n
- 3 ← liczba wystąpień litery i
- 2 ← liczba wystąpień litery e
- 1 ← liczba wystąpień litery b
- 1 ← liczba wystąpień litery s
- 1 ← liczba wystąpień litery k

- zmienna `char * text` oznacza wskaźnik do tablicy zawierającej wyraz
- ... oznacza, że należy samemu ustalić, jakie mają być pozostałe parametry wejściowe i wyjściowe

Funkcja `print_sequence` wyświetla w konsoli wartości wyliczonego ciągu.
Funkcja `recalculate_sequence` przelicza ciąg zgodnie z zasadą:

- pierwszy + ostatni
- (pierwszy + 1) + (ostatni -1)
- itd.

Dla ciągu wyznaczonego na podstawie słowa niebieski kolejne przekształcenia wyglądają następująco:
1 3 2 1 1 1
2 4 3
5 4

W przypadku, gdy w danym momencie sumowania w dowolnym ciągu/podciągu wynik sumowania jest większy od 10, np. $9 + 5 = 14$, to zapisujemy 1 4, czyli 1 i 4 na dwóch kolejnych pozycjach. Wykorzystując wyżej opisane funkcje napisz program, który pobiera od użytkownika wyraz nie dłuższy niż 100 znaków, zamienia go na ciąg, a następnie przelicza tak długo, aż otrzyma ciąg długości 2.

Przykładowe działanie programu:

```
Podaj słowo nie dłuższe niż 100 znaków:
alamakotakotmapsa
Wyznaczony ciąg liczb:
6 1 2 2 2 2 1 1
7 2 4 4
1 1 6
7 1
```

Zadanie 4 zgodność wyrazów

Korzystając z funkcji z zadania 3 napisz program, który przyjmie od użytkownika dwa ciągi znaków ze spacjami, połączy te ciągi i odrzuci spacje. Dla tak przygotowanego tekstu wyznaczy ciąg liczb i zredukuje go do dwóch liczb. Przykładowe działanie programu:

```
Podaj dwa napisy, każdy nie dłuższy niż 100 znaków:
napis pierwszy
napis drugi
napispierwszynapisdrugi
Wyznaczony ciąg liczb:
2 2 3 4 3 1 2 1 1 1 1 1 1
3 3 4 5 4 2 2
5 5 8 5
1 0 1 3
4 1
```