Woerter sortieren

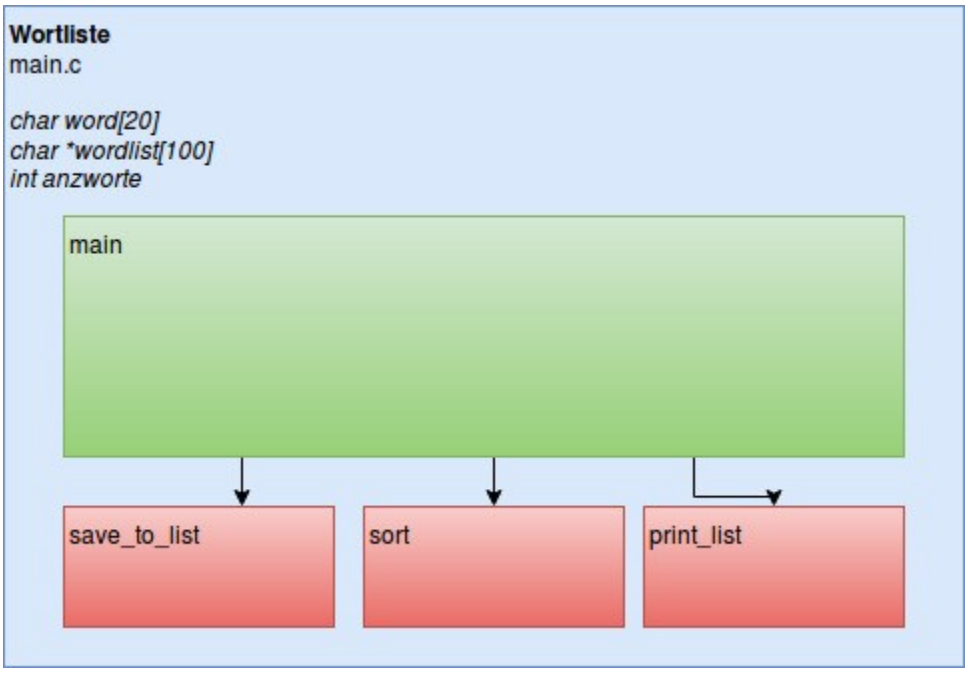
Zusammenfassung der Problemstellung

Das Programm soll ine Anzahl Wörter von der Tastatur einliest, diese in einem Array ablegt und zum Schluss alle Wörter alphabetisch sortiert ausgibt. Wiederholt eingegebene Wörter werden ignoriert und das Ende der Eingabe wird durch die Eingabe von „ZZZ“ markiert, d.h. danach werden die Wörter ausgegeben.

Lösungsvarianten

Die eingelesenen Wörter werden in einen Array von dynamisch allozierter Grösse gespeichert. Im Vergleich zu einem Array fixer Grösse sparen wir damit Platz.
In einer Liste speichern wir die Adressen aller eingelesenen Wörter. Der Vorteil daran ist, dass beim Sortieren keine Wörter sondern lediglich deren Pointer in der Liste geändert werden müssen.
Als Sortieralgorithmus verwenden wir Bubblesort. Da die Datenmenge relativ klein ist, fällt die Laufzeit nicht stark ins Gewicht und daher kann eine einfache Implementierung wie jene von Bubblesort verwendet werden.

Modulübersicht



Da es sich um ein relativ kleines Programm handelt, befindet sich der ganze Code in main.c, aufgeteilt auf die 4 Funktionen main, save_to_list, sort, print_list.

Globale Variablen:

- char word[20]

- char *wordlist[100]
- int anzworte

Lokale Variablen:
- char *wkopie (main)
- size_t listlaenge (main)
- int z (sort)
- char *temp (sort)
- int k (print_list)

Programmablauf

Main liest eine Anzahl von Wörtern ein, die Methode save_to_list speichert die Adresse der Worte in der Liste und sortiert sie anschliessend mit der Methode sort. Zum Schluss wird die geordnete Liste von der Methode print_list ausgegeben.

Tests

test_main_ausgabe: prüft, ob eine Anzahl eingegebene Wörter (inkl. mehrfach eingegebene Wörter) korrekt ausgegeben wird.

Alle Tests liefen erfolgreich durch.

Erkenntnisse

Das Programm läuft fehlerfrei.
Mögliche Erweiterungen: Anzahl vorkommen der Wörter zählen.

Anhang A: Quelltext

```
/* ---------------------------------------------------------------------------
 * --  _____       ______  _____                                             -
 * -- |_   _|     |  ____|/ ____|                                           -
 * --   | |  _ __ | |__   | (___    Institute of Embedded Systems            -
 * --   | | | '_ \|  __|  \___ \   Zuercher Hochschule Winterthur           -
 * --  _| |_| | | | |____ ____) |  (University of Applied Sciences)         -
 * -- |_____|_| |_|______|_____/   8401 Winterthur, Switzerland             -
 * ---------------------------------------------------------------------------
 */
/**
 * @file
 * @brief Lab implementation
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char word[20];
char *wordlist[100];
```

```
int anzworte = 0;

void save_to_list(char *wkopie);
void sort(char **wordlist);
void print_list(char **wordlist);

/**
 * @brief Main entry point. Reads several Words as input, finish input
 * with 'ZZZ'.
 * @returns Returns EXIT_SUCCESS (=0) on success,
 *                  EXIT_FAILURE (=1) on failure.
 */
int main(void)
{
   char *wkopie;
   size_t listlaenge;
   printf("Gib Woerter ein:\n");
   while(anzworte < 100 && !(strlen(word) == 3 && word[0]=='Z'
                        && word[1]=='Z' && word[2]=='Z')) {
      //read word
      scanf("%s", word);
      listlaenge = strlen(word) + 1;
      //generate dynamic an array with the right size
      wkopie = malloc(sizeof(char)*listlaenge);
      strcpy(wkopie, word);
      //safe word in wordlist
      save_to_list(wkopie);
      anzworte++;
   }
   anzworte--;
   //make alphabetic order
   sort(wordlist);
   //print out wordlist
   print_list(wordlist);
   return EXIT_SUCCESS;
}

/*
** function to sort the words in wordlist
*/
void sort(char **wordlist) {
   for(int z = 0; z < anzworte; z++) {
      for(int wortzaehler = 0; wortzaehler < anzworte - 1; wortzaehler++) {
      //compare words
         //test if equal
         if(strcmp(*(wordlist + wortzaehler),
                  *(wordlist + (wortzaehler+1))) == 0) {
            //if equal: delete one of equal words and move rest of the array -1,
            //reduce arraysize by 1
            free(wordlist[wortzaehler]);
            for(int tmp = wortzaehler; tmp < anzworte -1; tmp++) {
               wordlist[tmp] = wordlist[tmp+1];
```

```
            }
            anzworte--;
        }
        if(strcmp(*(wordlist + wortzaehler),
                *(wordlist + (wortzaehler+1))) > 0) {
            //swap words
            char *temp = wordlist[wortzaehler];
            wordlist[wortzaehler] = wordlist[wortzaehler + 1];
            wordlist[wortzaehler + 1] = temp;
        }
    }
  }
}

/*
** function to print out the sorted words in wordlist
*/
void print_list(char **wordlist) {
   printf("Die Wortliste ist:\n");
   for(int k = 0; k < anzworte; k++) {
      printf("%s\n", wordlist[k]);
   }
}

/*
** function to save each word in wordlist
*/
void save_to_list(char *wkopie) {
      wordlist[anzworte] = wkopie;
}
```