

Teil 8: Dynamische Datenstrukturen

Nutzen Sie bitte ausschließlich die Programmierumgebung VSC

8.1 Speicherbereiche

Gegeben ist das folgende C Programm

```
#include <stdlib.h>
#define MEAS_MAX 20
float x=0;
float polynom (float a1, float a2, float x)
{
    static int a3=0;
    a3++;
    return a1 * x * x + a2 * x + a3;
}

int main()
{
    float coefficients[] = {1.5, 12};
    float step = 1.0/MEAS_MAX;
    float *measValues = (float *) malloc (MEAS_MAX * sizeof(float));
    for (int i = 0; i < MEAS_MAX; i++)
    {
        measValues[i] = polynom (coefficients[0], coefficients[1], x);
        x += step;
    }
    return EXIT_SUCCESS;
}
```

- a) In welchen Speichersegmenten sind die Funktionen und Variablen abgelegt?
- b) Welche Gültigkeitsbereiche und Lebensdauern haben die Variablen?
- c) Was fehlt für den dynamisch reservierten Speicher?

8.2 Array dynamisch allokalieren für variable Größe

Reservieren Sie dynamisch ein Array für 10 Elemente vom Typ double. Initialisieren Sie die Werte mit zufälligen rationalen Werten zwischen 0 und 1.

Erweitern Sie das Array um 5 weitere Elemente und erzeugen 5 weitere zufällige Werte zwischen 1 und 2.

Anschließend geben Sie das Array aus, und geben den Speicherplatz wieder frei.

8.3 Dynamisches Array für Textanalyse

Der Benutzer gibt auf der Konsole einen Text ein, am Ende wird ^Z (Control-Z) eingegeben um den Text zu beenden. Der Text soll in ein dynamisches Array eingelesen werden. Das dynamische Array hat anfangs eine Kapazität für 10 Buchstaben. Wenn der Platz nicht ausreicht soll das Array jeweils um 10 weitere Elemente vergrößert werden.

Schreiben Sie Debug Logs in den Vergrößerungsschritten, damit nachvollziehbar ist, dass ihr Array korrekt vergrößert wird.

Erstellen Sie eine Funktion `numSentences()`, die als Parameter einen Pointer auf einen Text übergeben bekommt. Die Funktion überprüft in dem Text wie viele Sätze vorhanden sind. Der Einfachheit wird angenommen, dass es zu jedem Satz eines der folgenden Interpunktionszeichen gibt: '.', '?', '!'. D.h. die Funktion muss nur die Anzahl der Interpunktionszeichen zählen und als Rückgabewert zurückliefern.

Testen Sie die Funktion mit dem eingelesenen Text.

8.4 Dynamisches Array Debuggen

In folgendem Programm ist ein Fehler enthalten. Verwenden Sie den Debugger um den Inhalt der `buffer` Variablen nach jedem Einlesen auszuwerten und sehen sich die einzelnen Zeichen an.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAXLINE 100

int main()
{
    char *buffer;
    int len=0;
    buffer = malloc (MAXLINE);
    printf ("Bitte einen beliebigen Text eingeben und mit ^Z beenden.\n");
    while (fgets (buffer, MAXLINE, stdin))
    {
        len += strlen(buffer);
    }
    printf ("Der Text hat %d Zeichen.\n", len);
    printf ("Gesamte Eingabe: %s\n", buffer);

    free(buffer);
    return 0;
}
```