7.10.A Projekte/chkdsk-Monitor – Arbeitsblatt

1 Funktion und Testprogramm

In der unten stehenden Funktion `get_diskfree()` wird die Ausgabe von `df` (Linux) bzw. `chkdsk` (Wind.) dazu benutzt, die momentan zur Verfügung stehende Menge an Hintergrundspeicher einer Partition anzuzeigen. Ein erstes Programm benutzt diese Funktion.

- Ergänzen Sie die Funktion so, dass Fehler beim Aufruf von `system()` und `fopen()` mit einer Fehlermeldung auf dem Bildschirm *und* der Rückgabe eines negativen Wertes aus der Funktion gemeldet werden!
- Beachten Sie, dass alle in der Funktion `get_diskfree()` benutzten Zahlen per `#define` als Konstanten (anderer Name: Makros) festgelegt wurden und machen Sie es bitte auch so (gehört zum guten Ton in der strukturierten Programmierung)!

2 Erweiterungen nach Maß

Nun soll das Programm so erweitert werden, dass es mit Hilfe dieser Funktion für den Benutzer nützlich werden kann. Das neue Programm soll den Namen `getdf` erhalten.

- Welche Erweiterungen sind denkbar? Notieren Sie kurz in Stichpunkten!
- Programmieren Sie dabei möglichst strukturiert, d.h. benutzen Sie überall, wo es Ihnen sinnvoll scheint, Funktionen (und ggf. Datenstrukturen).
- Realisieren Sie eine Erweiterung nach der anderen. Benennen Sie dabei jede Version mit einer neuen Nummer, um den Überblick zu behalten!

```
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#define LINUX 0

long int get_diskfree(void);

int main(void)
{
   long int datenmenge;
   datenmenge=get_diskfree();
   printf("Freier Speicherplatz: %li kibiBytes\n", datenmenge);
   return 0;
}

#if LINUX
#   define BEFEHLSZEILE "df > tmp.txt"
#   define DATEI "tmp.txt"
#   define VON 120
#   define BIS 128
#   define FAKTOR 1
#   define DIVISOR 1
#else
#   define BEFEHLSZEILE "chkdsk c: /c /i > tmp.txt"
#   define DATEI "tmp.txt"
#   define VON 12065
```

```
#   define BIS 12075
#   define FAKTOR 1
#   define DIVISOR 1
#endif

long int get_diskfree(void)
{
   int c, lauf, rc;
   FILE *f;
   char zeichenkette[BIS-VON+2];

   system(BEFEHLSZEILE);
   f=fopen(DATEI,"r");

   for(c=lauf=0; lauf<VON && c!=EOF; ++lauf) c=fgetc(f);
   for(;lauf<BIS+1 && c!=EOF; ++lauf)
   {
      c=fgetc(f);
      if(c!=EOF) zeichenkette[lauf-VON]=c;
   }
   if(lauf>=VON)
      zeichenkette[lauf-VON]='\0';
   return atol(zeichenkette)*FAKTOR/DIVISOR;
   fclose(f);
}
```