Dr. Andreas Borchert   Institut für Angewandte Informationsverarbeitung   14. Dezember 2009
Wolfgang Kaifler / Michael Mattes   Blatt 10

Systemnahe Software I
im Wintersemester 2009/2010

Abgabetermin: bis 22. Dezember 2009

Makefiles

Ab sofort ist es für jedes Blatt notwendig, ein entsprechendes Makefile zu erstellen. Es sollen mindestens die einzelnen Programme erzeugt werden können sowie `make clean` zum Löschen von Objektdateien/Coredumps und `make realclean` zum Löschen der ausführbaren Dateien vorhanden sein.

Die Verwendung von Makefiles kommt auch Ihnen selbst zugute, denn so können Sie auf das Kompilieren mit Einzelbefehlen verzichten und müssen nicht daran denken, alle nötigen Include-Verzeichnisse, Compiler-Optionen etc. anzugeben. Durch das unkomplizierte Erstellen von ausführbaren Dateien sind Sie auch in der Lage, Ihr Programm schneller zu testen.

23 Zeilen einlesen mit stralloc (10 Punkte)

In dieser Aufgabe sollen Sie sich mit der `libowfat` vertraut machen und mit Hilfe der darin enthaltenen dynamischen Strings (die `stralloc`-Verbünde) ein Modul `saio.o` schreiben, das zwei Funktionen bereitstellt: eine Funktion zur Ausgabe eines `stralloc`-Strings auf der Standardeingabe und eine Funktion, die eine einzelne Zeile von der Standardeingabe liest und diese in einem gegebenen `stralloc`-String speichert.

Bieten Sie mit der Datei `saio.h` eine passende Programmierschnittstelle (API) zu Ihrem Modul an.

Beachten Sie außerdem folgendes: Das Newline-Zeichen soll beim Einlesen nicht im String enthalten sein. Das Ende der Eingabe wird wie ein Newline-Zeichen behandelt. Verwenden Sie bei der Implementierung ausschließlich I/O-Funktionen, die einzelne Zeichen lesen und schreiben! Das heißt, Sie sollen *keine der Funktionen* `read`, `scanf`, `printf` (oder deren Varianten wie z.B. `fread`) verwenden.

24 Editierdistanz (15 Punkte)

Schreiben Sie ein Modul `leven.o`, das eine einzelne Methode

```
int levenshtein(const char* w1, const char* w2);
```

anbietet. Der Rückgabewert dieser Funktion ist die Editierdistanz[1] der Worte `w1` und `w2`.

25 Soundex (15 Punkte)

Schreiben Sie das Modul `soundex.o`, welches die Methode

```
char* soundex(const char* w);
```

anbietet. Der Rückgabewert dieser Funktion ist der Soundex-Code[2] von `w`.

Die Funktion muss für den Rückgabewert Speicher reservieren, welcher vom Aufrufer der Funktion nach Benutzung wieder freigegeben werden muss.

26 Wortinfo (4 Punkte)

Schreiben Sie ein Programm `wordinfo.c`, welches interaktiv in jeder Zeile eine leerzeichengetrennte Liste von Worten entgegennimmt und dann bestimmte Informationen zu diesen Worten ausgibt. Erstens soll die paarweise Editierdistanz aller Worte in einer Matrix ausgegeben werden (achten Sie auf eine schön formatierte Ausgabematrix), zweitens soll zu jedem Wort der entsprechende Soundex-Code ausgegeben werden.

Verwenden Sie zum Einlesen der Worte Ihr Modul `saio.o` aus der ersten Aufgabe dieses Blatts. Für die Editierdistanz bzw. den Soundex-Code sollen Sie natürlich auch Ihre eben geschriebenen Module verwenden.

Viel Erfolg!

[1] Die Editierdistanz ist auch nach ihrem Erfinder als Levenshtein-Distanz bekannt. Weitere Informationen finden Sie unter `http://de.wikipedia.org/wiki/Levenshtein-Distanz`

[2] `http://de.wikipedia.org/wiki/Soundex`