Übungen zu Computergrundlagen WS 2010/2011

Übungsblatt 12: Einführung in C

18. Januar 2011

Allgemeine Hinweise

Abgabetermin für die Lösungen ist

- **Montag, 24.1., 13:00** für die Übungsgruppen am Mittwoch und Donnerstag
- **Donnerstag, 27.1., 13:00** für die Übungsgruppen am Montag und Dienstag

Die Lösungen solltest Du in eine Kopie der Datei `/share/Courses/CG2010/blatt12/blatt12.txt` einfügen, sofern sinnvoll. Zur Abgabe kannst Du entweder den Befehl

`/share/Courses/CG2010/bin/abgabe-<tutor> [<datei>...]`

ausführen (also z.B. `/share/Courses/CG2010/bin/abgabe-olaf blatt17.txt`), oder Du schickst die Datei(en) per Email an den jeweiligen Tutor.

Aufgabe 12.1: Berechnung von π (8 Punkte)

- 12.1.1 (je 2 Punkte pro Programm) Im Verzeichis `/share/Courses/CG2010/blatt12` befinden sich die Python-Skripte `pi1.py`, `pi2.py` und `pi3.py`, die auf drei verschiedene Weisen die Zahl π annähern (wie in Übungsblatt 4).

  Implementiere die drei Programme in der Programmiersprache C. Gib die fertigen C-Programm ab (den Quellcode, *nicht* das Binrprogramm!).

  Beachte dabei Folgendes:

  – Lass das C-Programm zehnmal so viele Punkte berechnen, wie das jeweilige Python-Programm.
  – Verwende den Typ `double` für Fließkommazahlen!
  – Zum Erzeugen einer Zufallszahl existiert die Funktion `rand()`. Diese erzeugt allerdings keine Fließkommazahl zwischen 0 und 1, sondern eine ganze Zahl zwischen 0 und `RAND_MAX`. Um eine Fließkommazufallszahl zwischen 0 und 1 zu erzeugen, musst Du also folgenden Befehl verwenden:

  ```
  double r = (double)rand() / RAND_MAX;
  ```

  Um die Funktion `rand()` verwenden zu können, muss die folgende Zeile am Anfang der C-Datei eingefügt werden:

  ```
  #include <stdlib.h>
  ```

  – Zum Berechnen der Wurzel einer Zahl existiert die Funktion `sqrt()`. Um sie (und andere mathematische Funktionen) verwenden zu können, muss die folgende Zeile am Anfang der C-Datei eingefügt werden:

  ```
  #include <math.h>
  ```

  Außerdem muss der Befehl zum Kompilieren die Option `-lm` verwenden.

– Der Gesamtbefehl zum Kompilieren von `pi1.c` sieht also so aus:

```
gcc -O3 -lm --std=c99 -o pi1 pi1.c
```

- 12.1.2 (2 Punkte) Vergleiche die Geschwindigkeiten der C-Programme und der Python-Programme. Trage die jeweiligen Laufzeiten in die Lösungsdatei ein.

  Wie viel mal schneller sind die C-Programme im Vergleich zu den entsprechenden Python-Programmen? Berücksichtige dabei die unterschiedliche Anzahl von Punkten (`MAX_STEPS`).

  Hinweis: Benutze wieder den Shellbefehl `time`, um die Laufzeit eines Programmes zu messen.

Aufgabe 12.2: Verständnis (2 Punkte)

- 12.2.1 (1 Punkt) Welche Vorteile bietet es, einen Interpreter zur Ausführung eines Programmes zu verwenden, anstatt es zu kompilieren?
- 12.2.2 (1 Punkt) Welchen Vorteil bietet es, ein Programm zu kompilieren, anstatt es mit Hilfe eines Interpreters auszuführen?

Hinweis: Verwechsle nicht die Eigenschaften der Programmiersprache damit, ob sie kompiliert wird oder interpretiert! Es wäre durchaus möglich, Python-Programme zu kompilieren, und es wäre auch möglich, einen Interpreter für C zu schreiben.