Programmieren in C++

SS 2011

Vorlesung 1, Mittwoch 4. Mai 2011

(Organisatorisches + ein erstes Programm)

Prof. Dr. Hannah Bast
Lehrstuhl für Algorithmen und Datenstrukturen
Institut für Informatik
Universität Freiburg

UNI FREIBURG

Blick über die Vorlesung heute

UNI FREIBURG

- Organisatorisches
  - Ablauf, Übungsblätter, Übungsgruppen, Regeln
  - Was Sie hier lernen sollen / Stil der Vorlesung
- Ein erstes Programm mit (fast) allem Drum und Dran
  - Berechnung der Wahrscheinlichkeit des Geburtstagsparadoxon
  - Makefile dazu schreiben
  - Unit Test dazu schreiben
  - Einen Linter drüber laufen lassen
  - Ergebnis ins SVN einchecken
  - Für das 1. Übungsblatt sollen Sie dann dasselbe machen für die Approximation von Pi ...

Ablauf 1 / 2

- Die Vorlesung
  - findet statt Mittwochs von 16.15 - 17.45 Uhr im HS 026
- Die Übungen …
  - sind der wichtigste Teil der Veranstaltung
  - Sie bekommen jede Woche ein Übungsblatt
  - Das können Sie machen wo und wann Sie wollen
  - Aber Sie müssen es selber machen!
  - Ich mache Ihnen jeweils vor was Sie brauchen
  - Wir fangen gleich heute damit an
- Die Note …
  - ergibt sich aus den Übungsblättern (10 Punkte / Blatt)
  - und einem kleinen Projekt am Ende

Ablauf 2 / 2

- Die Übungsgruppen ...
  - ... gibt es in diesem Sinne nicht; insbesondere können Sie die "offiziellen" Übungstermine (vom QIS) ignorieren
  - Es gibt aber Termine, zu denen Sie kommen können, wenn Sie Fragen / Probleme haben
  - Für das 1. Übungsblatt sind das:
    - Freitag, 6. Mai, 14 – 15 Uhr (Johanna und Markus)
    - Dienstag, 10. Mai, 15 – 16 Uhr (Mirko und Mathieu)
    - jeweils im Gebäude 51, Raum 00 030 (Erdgeschoss)
  - Die Tutoren für die Veranstaltung sind
    - Fabian Klein, Johanna Götz, Mathieu Wacker, Betim Musa, Markus Näther, und evtl. Mirko Brodesser, Marjan Celikik
  - Mailadressen und weitere Infos auf dem Wiki zum Kurs

Was mir wichtig ist

- Fragen Sie wenn etwas nicht klar ist
  - in der Vorlesung
  - über das Forum
  - Ihren Tutor
- Vorlesung
  - Ich würde Ihnen raten zu kommen
  - Aber sitzen Sie da nicht einfach nur passiv rum
  - Passen Sie auf, stellen Sie Fragen wenn etwas nicht klar ist, und Sie werden mit dem Übungsblatt nur die Hälfte der Arbeit haben
  - Die Vorlesungen werden aber auch aufgenommen
    - **Ich suche noch einen HiWi für das Schneiden !**

Was Sie hier lernen sollen

UNI FREIBURG

- Programmieren in C++ nach den Regeln der Kunst
  - im Umfang von 500 – 1000 Zeilen
  - Umsetzen von einfachen Lösungsideen in Programme
  - Grundlegende Programmkonstrukte in C++
  - Objektorientiertes Programmierem
  - Gutes Design, gute Namen, gute Dokumentation
  - Verständnis von Compiler und Linker
  - Benutzung eines Build Systems (make)
  - Unit Tests und Performance Tests
  - Benutzen eines Versionsverwaltungssystems (SVN)
  - Verwendung eines Stylesheets (cpplint.py)

Warum?

- Es gibt Programme ...
  - ... die lösen das gegebene Problem ... manchmal ... irgendwie
  - Zeitaufwand beim Erstellen: 1 h Programmieren, 10 h Fehlersuche
  - keiner außer dem Autor versteht das Programm
  - in einem Monat versteht es auch der Autor nicht mehr
  - jegliche Änderung / Erweiterung unmöglich, je größer das Programm desto unmöglicher
  - wenn man es nicht besser lernt, schreibt man solche Programme auch noch in 10 Jahren, und dann lernt man es nicht mehr
- Und dann gibt es Programme ...
  - ... die sieht man nach 6 Monaten oder länger wieder und freut sich, dass man auf Anhieb alles versteht
  - dann macht Programmieren Spaß, sonst nicht so

Zum Stil der Vorlesung

UNI FREIBURG

- Ich werde das meiste exemplarisch vormachen
  - Für die Details gibt es genügend Referenzmanuale
    - insbesondere in der Linux-Shell: man <Funktion>
  - Siehe auch die Referenzen auf der letzten Folie
  - Und Sie kennen ja Google und Co
  - Ich werde schauen, dass ich vor allem immer das erkläre was Sie auch gerade brauchen (für das nächste Übungsblatt)
- Fragen, Fragen, Fragen
  - Etwas ausprobieren, aber nicht zu lange, und dann fragen
    - aber nicht: "Mein Programm stürzt ab, warum?"
  - Die meisten Fragen interessieren auch andere, von daher vorzugsweise über das Forum (oder gleich in der Vorlesung)

Linux

- Wir entwickeln in diesem Kurs unter Linux
  - Falls Sie einen Windows-Rechner haben, haben wir ein Linux-Image (Virtual Box) für sie vorbereitet
    - Link zum Download auf dem Wiki zum Kurs
  - Wir arbeiten mit einem einfachen Editor und Makefiles, und von der Kommandozeile
    - So lernt man am besten was "under the hood" passiert
    - Aufwändige Entwicklungsumgebungen (Eclipse, NetBeans, Visual Whatever) sind was für die nächste Stufe

Unser Programm für heute

UNI FREIBURG

- Geburtstagsparadoxon
  - Was ist die Wahrscheinlichkeit, dass in einer Gruppe von n Leuten zwei am selben Tag Geburtstag haben?
    - unter der Annahme, dass die Geburtstage der Leute unabhängig und gleichverteilt über das Jahr sind
  - Die Wahrscheinlichkeit dafür ist

    $1 \cdot (364 / 365) \cdot (363 / 365) \cdot (362 / 365) \cdot \ldots$ (n mal)
  - Dafür schreiben wir jetzt ein C++ Programm nach den Regeln der Kunst
  - Für das erste Übungsblatt machen Sie dasselbe für die Approximation von Pi
    - für ein iteratives Verfahren Ihrer Wahl (das konvergiert)
    - zum Beispiel $\pi = 4 \cdot (1 - 1/3 + 1/5 - 1/7 + 1/9 \ldots)$

Make / Makefiles

- Ein sehr mächtiger Mechanismus
  - Wir benutzen es erstmal nur als eine Art Abkürzung für Befehle, die wir immer wieder verwenden
    - z.B. zum Kompilieren, Linten, Testen
  - Die Syntax im Makefile dafür ist einfach wie folgt

```
<target>:
        <Befehl 1>
        <Befehl 2>
        ...
```

Achtung: jede der Befehlszeilen muss mit einem TAB anfangen!

  - Wenn man dann (in dem Verzeichnis, in dem das Makefile steht) make <target> ausführt, werden einfach die entsprechenden Befehle ausgeführt
  - In den nächsten Vorlesungen mehr zu make ...

Google Test

UNI FREIBURG

- Google Test ist ein Unit Test Framework für C++
  - Unit Tests testen die Funktionalität einzelner Funktionen, typischerweise anhand von Spezialfällen
  - Ein guter Unit Test testet insbesondere
    - Randfälle, an denen potentiell etwas schief gehen kann
    - mindestens einen typischen allgemeinen Fall
  - Im Rahmen dieser Vorlesung werden wir fast immer sehr einfache Unit Tests schreiben
  - Trotzdem sind sie sehr nützlich …
    - … weil Sie Gewissheit geben, dass die einzelnen Funktionen im Prinzip richtig sind
    - … weil Sie zum Nachdenken zwingen, was eine Funktion eigentlich genau berechnen soll

Stylesheet / Lint

UNI FREIBURG

- Stylesheets sind wichtig
  - Nicht nur der Compiler muss Ihren Code verstehen, sondern auch andere Menschen
    - z.B. Ihr Tutor, ein Teamkollege (für spätere Projekte), oder Sie selber in drei Monaten …
  - Deswegen sehr wichtig sich an bestimmte Konventionen zu halten
    - manche Konventionen sind das Ergebnis langjähriger Programmiererfahrung, z.B. explicit (kommt später)
    - andere Konventionen sind einfach nur Standards um der Konsistenz willen, z.B. Einrückungstiefe
  - In Ihrer SVN Arbeitskopie liegt ein Skript cpplint.py, dass den korrekten Stil überprüft → soll ohne Fehler durchlaufen **!**

Daphne

- Unser neues Kursverwaltungssystem
  - Wurde über die letzten Monate von Jens Hoffmann und Axel Lehmann bei uns am Lehrstuhl entwickelt
  - Macht mir, den Tutoren, und hoffentlich auch Ihnen das Leben leichter
  - Registrieren Sie sich bitte nach der Vorlesung dort
    - dadurch bekommen Sie auch ein Unterverzeichnis (heißt so wie ihr RZ Username) in unserem SVN Repositorium → siehe nächste Folie
    - Bei Problemen Mail an Jens und Axel
      - E-Mail Adressen siehe Wiki zum Kurs

SVN

- SVN = Subversion
  - SVN ist ein sogenanntes Versionskontrollsystem
  - Es gibt ein sogenanntes Repositorium, das ist einfach ein Verzeichnisbaum mit Ordern drin, die liegen bei uns am Lehrstuhl auf einem Rechner
  - Jeder, der sich (via Daphne) bei uns registriert, hat ein Unterverzeichnis dort → URL siehe Ihre Daphne-Seite
  - Sie bekommen eine Kopie dieses Verzeichnisses mit

    svn checkout <URL> --username=<Ihr RZ Username>
  - In Ihrer Arbeitskopie können Sie dann Sachen ändern, Unterordner und Dateien hinzufügen, etc.
    - svn add <file name> fügt eine Datei erstmals hinzu
    - svn commit <file name> lädt die Änderungen zu uns hoch

Literatur / Links

- C++
  - http://www.cplusplus.com/doc/tutorial/
- Make
  - http://www.gnu.org/software/make/manual/
- SVN
  - http://subversion.apache.org/
- Google Test
  - http://code.google.com/p/googletest/

+ auf dem Wiki stehen kurze Einführungen dazu