ulm university universität
uulm

Labor Softwareentwurf mit Multiparadigmen-Programmiersprachen

Versuch 1
vom 20. Oktober 2014

H. Falk, A. Luppold

Bearbeitung bis: 2. November 2014

Allgemeines und Organisatorisches

Im Labor „Softwareentwurf mit Multiparadigmen-Programmiersprachen“ werden Ihnen die Vor- und Nachteile unterschiedlicher Programmierparadigmen anhand der Programmiersprache C++ vermittelt.

Im Laufe der Veranstaltung lernen Sie, Elemente der prozeduralen, objektorientierten, generischen und funktionalen Programmierung effektiv einzusetzen und zu kombinieren.

Die Veranstaltung legt dabei einen Schwerpunkt auf praktische Tätigkeiten im Rahmen aufeinander aufbauender Versuche. Vorkenntnisse in der Programmiersprache C++ sind ausdrücklich nicht notwendig. Der Umfang des Labors ist so ausgelegt, dass Sie sich parallel dazu in die Programmiersprache C++ einarbeiten können.

Bei Fragen zur Veranstaltung können Sie sich gerne an Ihren Übungsleiter wenden. Dies geht am einfachsten per E-Mail:

`arno.luppold@uni-ulm.de`

Organisatorisches:

Leider kommen auch wir nicht um die organisatorischen Details herum. Wir bitten Sie, diese aufmerksam zu lesen, da sie wichtige Informationen zur Veranstaltung enthalten.

- Die Organisation der Versuche erfolgt über die E-Learning-Plattform „**Moodle**“ der Universität Ulm (`https://moodle.uni-ulm.de/`).
- Die Versuche werden im Rahmen eines wöchentlich stattfindenden Präsenztermins im PC-Pool des Instituts für Eingebettete Systeme/Echtzeitsysteme (O27/311) bearbeitet. Der Termin wird zu Semesterbeginn festgelegt. Selbstverständlich können Sie auch selbstständig von zu Hause oder im PC-Pool des Instituts an den Versuchen arbeiten. Außerhalb der wöchentlichen Präsenztermine findet allerdings keine inhaltliche Betreuung statt.
- Die Versuche benötigen Vorbereitungszeit. Es wird erwartet, dass Sie sich im Vorfeld jedes Versuchs in die zu bearbeitende Thematik einarbeiten.

Durchführung und Bewertung des Labors

Die Veranstaltung besteht aus insgesamt 7 Versuchen, für die jeweils eine bestimmte Anzahl an Wochen vorgesehen ist. Ein Versuch besteht aus theoretischen Fragen zum Verständnis und praktisch zu implementierenden Aufgaben. Auch wenn wir natürlich den Austausch mit ihren Kommilitoninnen und Kommilitonen begrüßen, sollen Sie die einzelnen Versuche eigenständig bearbeiten.

Zu jedem Versuch muss eine schriftliche Auswertung erstellt werden. Diese besteht aus der Beantwortung der theoretischen Fragen, sowie der Auswertung der praktischen Aufgaben. Anleitungen zur Auswertung der Versuche werden innerhalb der Versuchsanleitung spezifiziert. Die Beantwortung der Fragen und die Auswertung müssen individuell von jedem Teilnehmenden selbst erstellt werden.

Die Modulprüfung besteht aus der Bearbeitung aller praktischen Aufgaben aller Versuche, sowie aus einer schriftlichen Auswertung in Papierform zu jedem Versuch. Die Modulprüfung ist sicher bestanden, wenn bei jedem einzelnen Versuch jeweils mindestens 50% der Gesamtpunkte erreicht wurden.

Der praktische Teil eines Versuchs wird gewertet, wenn Sie Ihren Quellcode über das Moodle-System abgegeben haben, die Funktion Ihres Programms Ihrem Betreuer vorgeführt haben und ihm die Funktionsweise erklären können. Stellen Sie sicher, dass alle eingereichten Dateien in UTF-8 codiert sind und Sie ausschließlich Unix-Zeilenumbrüche verwenden. Bei der Punktevergabe der praktischen Aufgaben werden neben funktionaler Korrektheit auch sauberer Programmierstil, wie beispielsweise Code-Einrückungen, klare Funktions- und Variablenbezeichnungen, sowie aussagekräftige Kommentare gewertet. Falsch codierte Dateien sowie unsaubere Abgaben (Abgaben die Binärcode,

___MACOSX-Verzeichnisse oder Ähnliches enthalten) führen zu Punktabzug. Für die Prüfung der Funktionalität ist entscheidend, ob der Quellcode auf den Compute-Servern des Instituts mit dem zur Verfügung gestellten GNU C++ Compiler fehlerfrei übersetzt werden kann. Erzeugen die Parameter −Wall −Wextra Warnmeldungen, so führt dies zu Punktabzug. Die Abgaben des praktischen Teils erfolgen über die Lernplattform Moodle. Der letztmögliche Abgabetermin für die praktischen Aufgaben ist auf jeder Versuchsbeschreibung angegeben.

Für den theoretischen Teil haben Sie die Möglichkeit eine Vorabversion Ihrer Ausarbeitung Ihrem Betreuer abzugeben, welcher Ihnen einmalig Tipps zur Verbesserung geben wird. Die Abgabe der Vorabversion muss innerhalb einer Woche nach dem letzten regulären wöchentlichen Termin des jeweiligen Versuchs geschehen.

Die endgültigen Versionen aller Ausarbeitungen müssen bis zum offiziellen Termin der Modulprüfung beim Betreuer in Papierform vorliegen. Der Ausarbeitung muss der vollständige Quellcode aller Versuche auf CD beigelegt sein.

Bei Einhaltung obiger Mindestanforderungen wird die Modulnote anhand der aufsummierten Gesamtpunkte über alle Theoriefragen und alle praktischen Aufgaben aller Versuche gebildet.

Bei der Bewertung gilt die Satzung der Universität Ulm zur Sicherung guter wissenschaftlicher Praxis. Bei ungenügender oder fehlender Referenzierung fremder oder auch eigener früherer Arbeiten wird ein Versuch mit 0 Punkten gewertet. Ebenso wird ein Versuch mit 0 Punkten bewertet, sollten Sie Quellcode oder Antworten zu den Theoriefragen von anderen Teilnehmern kopieren. Bei grobem Plagiarismus sowie bei Ausarbeitungen die nicht individuell erstellt wurden wird von einem Täuschungsversuch ausgegangen. Die Modulprüfung wird dann als „nicht ausreichend“ $(5,0)$ gewertet. Beachten Sie, dass die Rahmenprüfungsordnung der Universität Ulm in besonders schwerwiegenden Fällen oder bei wiederholter Täuschung auch den Ausschluss von einer Wiederholung der Modulprüfung mit der Folge des endgültigen Verlustes des Prüfungsanspruchs in dem betreffenden Studienfach vorsieht!

Nutzung des PC-Pools

Mit Ihrem Account können Sie die Rechnerinfrastruktur des Instituts nutzen. Es steht Ihnen der Rechnerpool mit den Betriebssystemen Windows oder Linux zu Verfügung. Das Passwort, das Sie bekommen haben, ist vorerst für alle Betriebssysteme gültig. Sie werden jedoch von Windows beim ersten Login aufgefordert, Ihr Passwort zu ändern. Da für Linux weiterhin Ihr initiales Passwort gesetzt ist, empfiehlt es sich, unter Linux das Passwort mit dem Befehl passwd ebenfalls zu ändern. Alternativ können Sie per SSH-Login auf `es-pwd.informatik.uni-ulm.de` beide Passwörter ändern.

Die Versuche im Rahmen dieses Labors werden auf den Compute-Servern des Instituts durchgeführt. Diese erreichen Sie per `ssh` unter den Adressen

- `wccsrv02.informatik.uni-ulm.de`, sowie
- `wccsrv03.informatik.uni-ulm.de`

Wenn Sie im PC-Pool Linux starten, erfolgt ihr Login automatisch auf dem Server `wccsrv02`. Sie können dann sofort damit beginnen Ihre Aufgaben zu bearbeiten. Möchten Sie lieber Windows nutzen, können Sie sich mit dem SSH-Client `PuTTY` auf einem der beiden Server anmelden.

Vermeiden Sie es möglichst, im PC-Pool unter Windows Daten in Eigene Dateien zu speichern. Verwenden Sie stattdessen Ihr Home-Verzeichnis (Netzlaufwerk `H:`). Das Netzlaufwerk `H:` entspricht ihrem Heimatverzeichnis unter Linux.

Versuchsvorbereitung und Literatur

Für die erfolgreiche Teilnahme an der Veranstaltung sind keine C++ Vorkenntnisse erforderlich. Die Veranstaltung selbst kann allerdings nicht als C++ Kurs dienen. Es wird daher erwartet, dass Sie sich mit dem Themengebiet jedes Versuchs selbstständig im Vorfeld der Präsenztermine beschäftigen. Bitte bedenken Sie, dass die Bearbeitungszeit der Aufgaben zu den Präsenzterminen nicht darauf ausgelegt ist, dass Sie sich während der Termine noch die notwendigen Grundlagen aneignen müssen. Als Ansatzpunkt für Ihre Vorbereitung können Ihnen die zu Beginn jedes Blattes gestellten Theoriefragen dienen. Sie sollten in der Lage sein, vor der Teilnahme am Präsenztermin diese Fragen zu beantworten.

Prinzipiell können Sie als Literatur ein beliebiges C++ Buch wählen. Wir begrüßen es ausdrücklich, wenn Sie sich für das Lehrbuch entscheiden, das Ihrem Lernstil am Besten entspricht. Innerhalb des Labors wird konsequent der neue C++ Standard C++11 verwendet. Sie sollten bei der Wahl eines Lehrbuchs darauf achten, dass dieser vor Allem von älteren Lehrbüchern oftmals nicht thematisiert wird.

Als Einstieg in die Programmiersprache C++ empfehlen wir Ihnen daher das Buch `C++ Primer` von Stanley B. Lippman in der Fünften Auflage. Das Buch ist in der Bibliothek der Universität verfügbar und wird von der Universität auch als E-Book angeboten. Zum E-Book gelangen Sie, indem Sie innerhalb des Uninetzwerkes `http://proquest.tech.safaribooksonline.de/` aufrufen und dort nach „C++ Primer“ suchen.

Als Online-Referenz eignen sich des Weiteren die Webseiten `http://www.cplusplus.com` und `http://www.cppreference.com`.

Erste Schritte, Prozedurale Programmierung I

Versuch 1-1: Anmeldung in Moodle (keine Punkte)

Alle Unterlagen und Informationen zu den Versuchen werden über die Lernplattform Moodle bereitgestellt.

Melden Sie sich in Moodle (`https://moodle.uni-ulm.de/`) mit Ihrem kiz-Account an. Treten Sie dann der Veranstaltung „Softwareentwurf mit Multiparadigmen-Programmiersprachen“ im Wintersemester 2014/2015 bei.

Versuch 1-2: Die Compute-Server (keine Punkte)

Loggen Sie sich zunächst per SSH auf einem der Server `wccsrv02` oder `wccsrv03` ein. Die am Institut eingesetzten Compute-Server bieten neben einer Standard Linux-Umgebung auch eine Vielzahl spezialisierter Softwareprogramme. Eine Liste der vorhandenen Module erhalten Sie mit dem Befehl `module avail`. Für diese Veranstaltung benötigen Sie ausschließlich die GNU Compiler Collection in der Version 4.8.1. Sie laden diese mit dem Befehl `module load gcc/4.8.1`.

a) Laden Sie die Compiler-Umgebung mit obenstehendem Befehl. Vergewissern Sie sich mithilfe des Befehls `g++ -v`, dass Sie tatsächlich Version 4.8.1 des C++ Compilers nutzen.

b) Um das Modul nicht nach jedem Login erneut manuell laden zu müssen, tragen Sie in die Datei `.modules.linux` in Ihrem Heimatverzeichnis die folgende Zeile ein:

```
1 +gcc/4.8.1
```

Melden Sie sich ab und wieder an, oder öffnen Sie ein neues Terminal und stellen Sie sicher, dass der GCC 4.8.1 automatisch geladen wird.

Versuch 1-3: Theorie (18 Punkte)

a) Machen Sie sich mit den folgenden Programmierparadigmen vertraut:

- Prozedurale Programmierung
- Objektorientierte Programmierung
- Generische Programmierung
- Funktionale Programmierung

Erklären Sie jeweils kurz, was man unter den jeweiligen Punkten versteht.

b) Erklären Sie die Bedeutung der Schlüsselwörter struct und const in C++.

c) Was ist ein Präprozessor? Was versteht man in C++ unter Macros, und wofür werden sie benötigt? Welche Probleme birgt der Einsatz von Macros?

d) Erläutern Sie den Unterschied zwischen Pointern und Referenzen.

e) Erklären Sie die einzelnen Verarbeitungsschritte, die Ihr Quellcode beim Compilieren mit dem `g++` vom Einlesen des Programmcodes bis hin zum Erstellen der fertig gelinkten Programmdatei durchläuft.

Versuch 1-4: Ihr erstes C++ Programm / Übersetzen von Programmen (5 Punkte)

a) Tippen Sie den folgenden kurzen Quellcode ab:

```
1  #include <iostream>
2
3  int main() {
4    std::cout << "Hello World!" << std::endl;
5    return 0;
6  }
```

Speichern Sie den Code unter dem Dateinamen `hello-world.cc`. Beachten Sie dabei die Konvention, dass in C++ Programmcode-Dateien die Endung `.cc` oder `.cpp` erhalten.

Sie können Ihr erstes Programm nun übersetzen. Wechseln Sie dazu auf der Kommandozeile in das Verzeichnis mit Ihrem Quellcode und übersetzen Sie das Programm mit dem Befehl

```
1 g++ -std=c++11 -Wall -Wextra -o hello.out hello-world.cc
```

b) Erläutern Sie die einzelnen Parameter, die dem Compiler übergeben werden.

Versuch 1-5: Ein persönlicheres Hello World (5 Punkte)

a) Erweitern Sie Ihr Programm so, dass es Ihren Namen von der Standardeingabe (stdin) lesen und auf der Standardausgabe (stdout) ausgeben kann.

b) Schreiben Sie Ihr Programm so um, dass Sie Ihren Namen beim Programmaufruf als Parameter übergeben können.

Versuch 1-6: Ackermann-Funktion (5 Punkte)

Machen Sie sich mit der Ackermann-Funktion vertraut. Schreiben Sie ein prozedural aufgebautes Programm, das über die Standardeingabe (stdin) die Parameter für die Funktion einliest, und über die Standardausgabe (stdout) das Ergebnis ausgibt. Bei ungültigen Argumenten soll eine Ausgabe auf die Standardfehlerausgabe (stderr) erfolgen.

Versuch 1-7: Bibliotheken (10 Punkte)

Schreiben Sie ein Programm, das einfache Berechnungen ermöglicht. Das Programm soll über die Standardeingabe (stdin) den Namen einer zu berechnenden Funktion sowie den zu übergebenden Wert bekommen, und das Ergebnis auf der Standardausgabe (stdout) ausgeben. Dabei sollen die Funktionen sin(), sqrt() sowie tanh() unterstützt werden. Ein Beispielprogramm könnte so aussehen:

```
1 $ ./calc.out
2 Was moechten Sie berechnen? sin 1.0
3 Ergebnis: 0.84147
4 $
```

Sowohl die Verarbeitung der Eingabedaten als auch die Berechnungen sollen dabei in separaten Funktionen ausgeführt werden.

Erläutern Sie, was beim Schreiben Ihres Programms beachten müssen, damit die mathematischen Funktionen beim Übersetzen gefunden werden.

Versuch 1-8: Pointer und Referenzen (10 Punkte)

a) Schreiben Sie ein Programm, das eine Methode enthält die den Inhalt zweier übergebener integer-Variablen vertauscht.

   - Implementieren Sie Ihr Programm mithilfe von Pointern.
   - Nutzen Sie in in einem zweiten Versuch Referenzen statt Pointer.

   Welche Unterschiede, welche Vor- und Nachteile erkennen Sie bei beiden Methoden?

b) Scheiben Sie ein Programm, das eine Funktion print() enthält. Diese soll ein gegebenes Integer-Array traversieren und dessen Inhalt auf die Standardausgabe ausgeben. Implementieren Sie die folgenden Versionen von print():

   - void print1(int*);
   - void print2(int[]);

- `void print3( int[20]);`
- `void print4( int (&array)[10]);`

Schreiben Sie ein Testprogramm, das die vier Funktionen der Reihe nach aufruft.

Welche Probleme und Einschränkungen bestehen bei den Implementierungen? Wie lassen sie sich beheben? Existieren Unterschiede zwischen den einzelnen Aufrufen? Falls ja, welche?