Hochschule RheinMain
Fachbereich Design Informatik Medien
Marcus Thoss, M.Sc.

Hardwarenahe Programmierung II
SS 2020
LV 2512

Übungsblatt 1

Aufgabe 1.1 (Entwicklungsumgebung):

Für die Bearbeitung der Praktikumsaufgaben wird im weiteren Verlauf u.a. die integrierte C++-Entwicklungsumgebung *NetBeans*, der UML-Diagrammgenerator *PlantUML* und natürlich die gcc-Compiler-Suite (nun mit `g++` als Frontend) verwendet. Richten Sie sich diese für Ihren Account auf dem Praktikums-PC und in Ihrer privaten Arbeitsumgebung (z.B. Laptop) ein.

a) Für den C++-Compiler ist die genaue Version für unsere Zwecke zweitrangig. Stellen Sie sicher, dass das Tool `g++` in der Shell verfügbar ist, anderenfalls instllieren Sie den Compiler (z.B. einfach `apt install g++`).

b) PlantUML ist ebenfalls auf den Pool-PCs verfügbar; installieren können Sie es privat als Paket (sofern verügbar, Ubuntu: Paket plantuml) oder einfach durch Download von plantuml.jar von `http://plantuml.com/` (nicht https://).

   Sie können PlantUML durch `plantuml -testdot` (bzw. `java -jar plantuml.jar -testdot`) auf korrekte Installation prüfen.

c) Mit der integrierte Entwicklungsumgebung NetBeans befassen Sie sich später, zunächst ist es, wie zuvor in HWP1, wichtiger, sich mit dem Build-Prozess durch Ausführen der Build-Schritte im Detail vertraut zu machen. Sie sollten aber die Installation schon durchführen. Verwenden Sie bitte die Netbeans-Version 11.0 (LTS, s. `https://netbeans.apache.org/download/nb110/nb110.html`) und beachten Sie die Hinweise unter `https://netbeans.apache.org/download/nb113/index.html#_notes`, denn Sie müssen die Netbeans-C++-Plugins aus Version 8.2 integrieren.

   JDK-Installation

   Für das Compilieren und Ausführen von Java-Anwendungen (und damit auf für Netbeans) benötigen Sie eine Java-Build-Umgebung. Diese wird im Java-Sprachgebrauch als *JDK* (Java Development Kit) bezeichnet. Für die Verwendung von Netbeans wäre ein *JRE* (Java Runtime Environment) ausreichen, aber damit erhalten Sie nur

Tools zum Ausführen von Java-Anwendungen und *keinen* Compiler für die eigene Programmentwicklung, was u.a. für die LV ADS erforderlich ist.

Die Entwicklung sollte wie immer möglichst unter Linux stattfinden, damit Sie weiterhin Linux-fit bleiben. Prinzipiell können Sie Java-basierte Entwicklung auch unter Windows oder MacOS erledigen, müssen sich dann aber ggf. selbst um Support bei Problemen kümmern, und die spätere C++-Entwicklung wird ohnehin Linux erfordern.

Ebenfalls wichtig: Verwenden Sie für den Kurs die JDK-Version 8 oder 11 des *OpenJDK*. Unter Ubuntu ist die Installation mit `sudo apt install openjdk-8-jdk` möglich. Wenn Sie Java in Version 8 oder 11 nicht über die Installationstools Ihrer Betriebssystem-Umgebung erhalten können, können Sie sich auch das Oracle-JDK von `https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html` herunterladen.

Der Grund für die Verwendung dieser JDK-Versionen ist u.a. die Kompatibilität mit der grafischen NetBeans-Entwicklungsumgebung, die wir uns im Laufe des Kurses auch anschauen wollen. Installieren Sie also bitte nicht einfach die neueste Java-Version, oder installieren Sie zumindest *zusätzlich* eine der genannten Versionen.

Um die Version der Installieren Laufzeitumgebung zu prüfen, geben Sie

```
java -version
```

ein. Für Version 8 müsste dann ungefähr folgende Ausgabe erfolgen:

```
"openjdk version 1.8.0_171"
```

Das Starten der Netbeans-Umgebung erfolgt unter Linux über das Skript `netbeans` im `bin/`-Unterverzeichnis Ihrer Netbeans-Installation. Bei Problemen mit der Java-Runtime-Umgebung beachten Sie, dass Sie in der Datei `/etc/netbeans.conf` die zu verwendende JDK-Version festlegen können.

Aufgabe 1.2 (Das erste C++-Programm):

a) Legen Sie in einem Übungsverzeichnis eine Textdatei `hwp2_p1.cpp` an und schreiben Sie darin ein *Hello-World*-Programm unter Nutzung von `cout` wie in der Vorlesung gezeigt.

b) Compilieren Sie das Programm mit `g++`. Die Ihnen von `gcc` bekannten Flags wie `-Wall` oder `-c` und `-o` funktionieren hier genauso, nutzen Sie sie! Führen Sie das erzeugte Programm testweise aus.

c) Erstellen Sie ein Makefile, um die Übersetzung zu automatisieren.

d) Versionieren Sie Ihre Quellen mit `git`.

e) Erweitern Sie das Programm so, dass zwei Zahlen eingegeben, in Variablen abgelegt und addiert werden und das Ergebnis hexadezimal ausgegebn wird. Nutzen Sie dazu `std::cin` und `std::hex`.

Aufgabe 1.3 (Das erste UML-Diagramm):

a) Erstellen Sie in Ihrem Übungsverzeichnis eine Datei `animal.txt`, in der die Klasse `Animal` für die Darstellung mit PlantUML beschrieben wird.

   Die Klasse soll die Eigenschaft `weight` (ein `float`) und die Methoden `feed()`, die ein `float` als Parameter erhält und nichts zurückgibt, sowie `getWeight()` (ohne Parameter, Rückgabetyp `float`) besitzen.

b) Erzeugen Sie mit PlantUML aus der Beschreibungsdatei jeweils eine PNG- und eine PDF-Datei, die die Klasse als UML-Diagramm zeigt.