Übungsblatt 7
Propädeutikum Informatik

June 9, 2018

Dieses Übungsblatt wird bewertet und bildet die Modulnote für das Propädeutikum Informatik im Sommersemester 2018. Alle Rechnungen sind mit Rechenweg anzugeben. Aufgabe 1,2,3 sind als pdf Datei abzugeben und die Programmieraufgaben in .java Dateien. Die komplette Abgabe ist in einem zip-Archiv bis zum 22.06.2018 23:55 Uhr an fm63byza@studserv.uni-leipzig.de zu senden. Sämtliche Klassen und Methoden, die wir im Propädeutikum erarbeitet haben können genutzt werden. Für jede Programmieraufgabe ist mindestens ein Beispielaufruf in der Main Mathode anzugeben.

1 Umrechnung zwischen Zahlensystemen

Rechne die folgenden Zahlen vom Dezimal- in das Dualsystem um: 42 und -42. Stelle die Zahlen dabei wie in der Übung als 8-Bit Dualzahlen dar.

Rechne die folgene Dualzahl in das Dezimalsystem um: 111,11.

2 Binäre Suchbäume

Gegeben sei folgende Liste L=[9,5,11,2,1,8,6,10,3,15]. Baue aus L einen Binären Suchbaum auf.

3 Erstellung eines UML-Diagramms

Erstelle ein UML-Diagramm zu folgendem Sachverhalt:

Modelliert werden soll ein Fußballspiel. Eine Person besitzt einen Namen und ein Geburtsjahr. Ein Fußballspieler ist eine Person mit den zusätzlichen Eigenschaften einer Trikotnummer und einer Anzahl bisher geschossener Tore. Eine Fußballmanschaft besitzt einen Namen ein Gründungsjahr, ein Budget, einen Trainer vom Typ "Person" und 11 Fußballspieler die in einem Array gespeichert werden sollen. Ein Fußballspiel besteht aus einer Heimmanschaft und einer Gästemanschaft, einem Schiedsrichter vom Typ Person, und der Anzahl an Heim- und Gästetoren. Die Klasse Fußballspiel soll außerdem über eine Methode Verfügen, welche die Gesamttorzahl zurückgibt und eine Methode, welche die Gewinnermanschaft zurückgibt.

Auf die Angabe von Konstruktoren kann im UML verzichtet werden, die Getter und Setter Methoden für private Attribute sollen aber in jeder Klasse mit angegeben werden. Beachte, dass nur ein UML Diagramm zu erstellen ist. Eine implementierung in Java ist nicht notwedig.

4 Collatz-Folge

Schriebe ein Methode collatzNachfolger, welche für einen gegebenen Integer Wert den Nachfolger nach der folgenden Rechenvorschrift[1] berechnet:

$collatz(n) = n/2$, falls n gerade ist
$collatz(n) = (3 * n) + 1$, falls n ungerade ist.

Schreibe außerdem eine Methode collatzIteration, welche für alle Zahlen zwischen 2 und 100 berechnet, wie oft die Collatz Funktion angewandt werden muss, bis das Ergebnis 1 wird. Das Ergebnis soll für jede Zahl zwischen 2 und 100 auf dem Bildschirm ausgegeben werden und in einem Interger-Array gespeichert und von der Methode zurückgegeben werden.

5 gewichtete Summe

Schreibe eine Java Methode, die die gewichtete Summe eines Integer Array zurück gibt. Die Methode soll ein Interger Array als Parameter nehmen und die Summe als Rückgabewert haben. Die gewichtete Summe ist die Summe, bei der jedes Array-Element vor dem summieren mit seinem Index multipliziert wird. Also ErstesElemet*1+ZweitesElement*2 usw. .

6 showZins

Erweitere die Klasse Sparbuch aus der Übung um eine Methode schowZins(), welche für das gegebene Guthaben und den gegebenen Zinssatz auf dem Bildschirm ausgibt, wie viele Zinsen man nach 1,2,5 bzw. 10 Jahren bekommen würde.

7 Zahlen sortieren

Schreibe eine Methode sort(), welche ein Integer Array übergeben bekommt und das Array sortiert wieder zurückgibt. Der verwendete Algorithmus kann selber gewählt werden. Es kann einer der im Propädeutikum besprochen sein oder ein Algorithmus selber entworfen werden. Zeit- und Speicherkomplexität sind ebenfalls irrelevant. Es kann davon ausgegangen werden, dass in dem Ausgangsarray keine Elemente mehrfach vorkommen.

[1] `https://en.wikipedia.org/wiki/Collatz_conjecture`