

Klausur A
Einführung in die Programmierung
10.03.2021 (WS 2020/21)

Hinweise

1. Bearbeitungszeit: 60 Minuten
2. Zum Bestehen der Klausur sind 50 Punkte erforderlich.
3. Erlaubte Hilfsmittel: keine
4. Notieren Sie Ihre Lösungen auf eigenem Papier. Versehen Sie jedes Blatt mit Namen, Matrikelnummer, Studiengang und Seitenzahl. **Geben Sie bei jeder Lösung die Aufgabennummer und Teilaufgabe an.** Bitte geben Sie nicht die Lösungen von zwei verschiedenen Aufgaben auf einem Blatt an.
5. Bitte schreiben Sie deutlich mit einem schwarzen Stift.
6. Alle vorgegebenen und zu erstellenden Programmtexte beziehen sich auf die Programmiersprache Java.

Viel Erfolg!

Aufgabe	Max. Punktzahl	Erreichte Punktzahl
1 (Zahlendarstellung)	7	
2 (Ein-/Ausgabe, Programmieraufgabe)	31	
3 (Array-Programmierung)	31	
4 (UML nach Java übertragen)	31	
<i>Bonuspunkte</i>	<i>10</i>	
Gesamtpunktzahl	100+10=110	

Aufgabe 1 (Zahlendarstellung)**[7 Punkte]**

Beschreiben Sie am Beispiel der Zahl $n = 47$, wie man die Hexadezimaldarstellung einer Zahl berechnet.

Aufgabe 2 (Ein-/Ausgabe, Programmieraufgabe)**[31 Punkte]**

Implementieren Sie eine Klasse **ZeichenFinden** mit den folgenden zwei Methoden:

i) **String löscheErstesVorkommen(String w, char c)**

Es wird in dem Wort **w** das erste Vorkommen des Zeichens **c** gelöscht und das resultierende Wort zurückgegeben. Sollte **c** nicht in **w** vorkommen, so geben Sie einfach **w** zurück.

Bspl.: Wird für **w** das Wort „abbA“ und für **c** das Zeichen „b“ übergeben, so soll das Wort „abA“ zurückgegeben werden.

ii) **String löscheAlleVorkommen(String w, char c)**

Es werden in dem Wort **w** alle Vorkommen des Zeichens **c** gelöscht und das resultierende Wort zurückgegeben.

Bspl.: Wird für **w** das Wort „abbA“ und für **c** das Zeichen „b“ gewählt, so soll das Wort „aA“ zurückgegeben werden.

Sie dürfen aus der Java-Bibliothek nur die Methoden `length()` und `charAt()` auf Strings benutzen. Für einen String **w** gibt die Methode `w.length()` die Länge von **w** und die Methode `w.charAt(i)` das *i*-te Zeichen von **w** zurück.

Aufgabe 3 (Array-Programmierung)**[31 Punkte]**

In dieser Aufgabe soll eine Klasse bereitgestellt werden, in der für 7 Tage Temperaturwerte gespeichert und ausgewertet werden sollen. Hierbei sind für jeden Tag mehrere Temperaturwerte möglich. Die Werte sollen in einem 2-dimensionalen Array mit 7 Zeilen gespeichert werden (eine Zeile pro Tag).

Implementieren Sie die folgenden Methoden:

i) `double[][] legeFeldAn()`: Es soll das Array mit 7 Zeilen angelegt und zurückgegeben werden, in dem noch keine Werte enthalten sind.

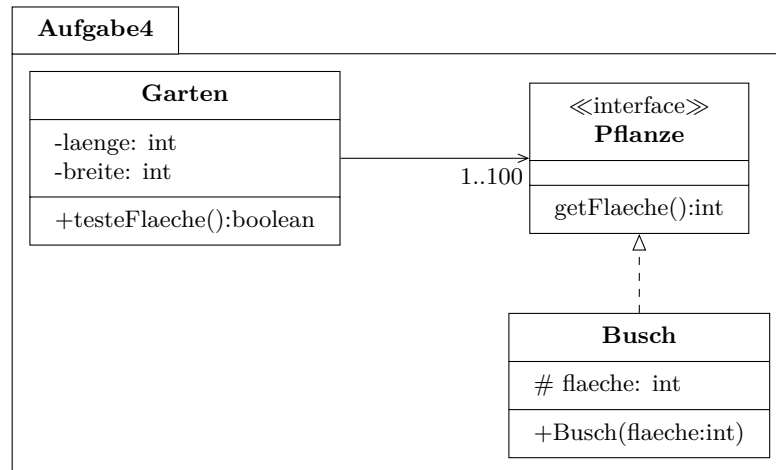
ii) `double[][] fuegeWertHinzu(double[][] feld, int tag, double wert)`: Es soll für den Tag **tag** der Wert **wert** hinzugefügt werden. Sollten **feld** oder **tag** fehlerhaft sein, d.h. $\text{tag} < 0$ oder $\text{tag} \geq 7$ bzw. **feld** hat nicht 7 Zeilen, so geben Sie die null-Referenz zurück.

iii) `double berechneDurchschnitt(double[][] feld, int tag)`: Es soll für den Tag **tag** die Durchschnittstemperatur ausgerechnet werden. Falls die Parameter fehlerhaft sind, so darf hier ein beliebiger Wert zurückgegeben werden.

Sie dürfen **keine** Klassen- oder Objektattribute oder Methoden aus der Java-Klassenbibliothek benutzen. Sie können aber weitere eigene (Hilfs-)Methoden hinzufügen.

Aufgabe 4 (UML nach Java übertragen)**[31 Punkte]**

Gegeben sei das folgende UML-Klassendiagramm. Programmieren Sie die angegebenen Klassen und Schnittstellen vollständig in Java.



Hinweis: Es dürfen **keine** Klassen der Java-Klassenbibliothek verwendet werden!

Ergänzen Sie notwendige Attribute und Methoden, insbesondere sinnvolle Konstruktoren und alle Attribute und Methoden zur Verwaltung der Assoziation.

Die Methode `testeFlaeche()` soll überprüfen, ob die Summe der Flächen (`getFlaeche()`) aller assoziierten Pflanzen kleiner oder gleich der Fläche des Gartens (Länge mal Breite des Gartens) ist. Wenn das der Fall ist, so soll `true`, sonst `false` zurückgegeben werden.