Prüfung
Computer Programming (CP)

Prof. Dr.-Ing. Christian Heller <christian.heller@ba-leipzig.de>

Student	
Vor- und Nachname	Ihre Daten werden von der Klausuraufsichtsperson schriftlich auf der Anwesenheitsliste festgehalten.
Matrikelnummer	
Studienrichtung und Jahr	CS 2018-1
Anmeldename	Das Login "klaus..." muss unbedingt schriftlich auf der Anwesenheitsliste festgehalten werden, da sonst keine Zuordnung des Logins zu Ihrem Namen und damit keine Korrektur der Klausur möglich ist!

Prüfung	
Datum	Dezember 2018
Dauer [min]	120
Hilfsmittel	Dokumentation im lokalen Netzwerk (Intranet) sowie Recherche im Internet. NICHT gestattet: * Kommunikation in jeglicher Form * Anmeldung via SSH auf dem Rechner "fileserv" * Anmeldung mit Klausur-Login nach Ende der Prüfung Dies kann leicht geprüft werden (last cs16*, Server-Log-Dateien). Bitte unterlassen Sie also Täuschungsversuche in Ihrem eigenen Interesse.
Bemerkungen	Hinterlegen Sie alle Programme und Antworten in elektronischer Form! Es wird kein Papier angenommen. Möchten Sie Lösungen erläutern, so nutzen Sie Quelltext-Kommentare oder legen eine Text-Datei an. Speichern Sie sämtliche Daten im HOME-Verzeichnis des Nutzers, d. h. unter Windows auf Laufwerk H:\ (NICHT auf C:\ oder "Eigene Dateien")! Idealerweise legen Sie dort ein Unterverzeichnis namens "klausur" an. Lesen Sie die Aufgaben komplett durch, bevor Sie sie lösen! Die Reihenfolge der Lösung ist Ihnen überlassen. Probieren Sie immer, eine Aufgabe zu lösen, da auch auf richtige Teile nicht vollständiger Lösungen Punkte vergeben werden! Falls vom Prinzip her richtig, so werden auch alternative Lösungen akzeptiert. Sie dürfen beliebig viele Bildschirmausgaben von Werten in den Quelltext einbauen, um ein Programm besser nachvollziehen zu können. Bitte duplizieren Sie Ihre Quelltextdateien (workspace) NICHT, da beim Korrigieren dann beide durchsucht werden müssen, was sinnlosen Aufwand verursacht. Diese Aufgabenstellung in Papierform können Sie nach dem Ende der Klausur behalten.

Bewertung						
Aufgabe	1	2	3	4	5	Summe
Punkte	20	30	10	20	20	100

Ihre Programmierkenntnisse werden anhand kleiner, voneinander unabhängiger Java-Programme geprüft.

Aufgabe 1: Structure [20]

Zweck: Bestimmen eines Schaltjahres.

a) Deklarieren Sie in der Startmethode eine lokale Variable namens „jahr“ vom Typ „int“! [2]

b) Weisen Sie ihr eine beliebige Jahreszahl als Initialwert zu! [2]

c) Prüfen Sie, ob genau ein Kommandozeilenargument angegeben wurde! [2]

d) Weisen Sie seinen Wert im positiven Falle der lokalen Variablen „jahr“ zu! [2]

Hinweis: Konvertierung in Ganzzahl nötig.

e) Definieren Sie außerdem eine lokale „boolean“-Variable namens „schaltjahr“ mit dem

Initialwert „false“! [2]

f) Prüfen Sie, ob sich das Jahr durch 4 teilen lässt! Falls nicht, so ist es KEIN Schaltjahr. [2]

Hinweis: Modulo-Operator

g) Prüfen Sie im positiven Falle, ob es sich auch durch 100 teilen lässt! Falls NICHT, so IST es ein Schaltjahr. [2]

h) Prüfen Sie im positiven Falle, ob es sich auch durch 400 teilen lässt! Falls ja, so IST es ein Schaltjahr; anderenfalls nicht. [2]

i) Setzen Sie den Wert der Variablen „schaltjahr“ für alle drei eben durchgeführten Fallunterscheidungen entsprechend auf „true“ oder „false“! [2]

j) Geben Sie den Wert der Variablen „schaltjahr“ als Ergebnis auf der Konsole aus! [2]

Ergebnis: Auf der Konsole wird ausgegeben, ob es sich um ein Schaltjahr handelt.

Aufgabe 2: Array [30]

Zweck: Ausgeben eines Wortes als 3D-ASCII-Art.

Gegeben sei eine Klasse namens „Alphabet“ mit einem zweidimensionalen „String“-Array.

a) Erstellen Sie eine Startklasse mit „main“-Methode! [2]

b) Deklarieren Sie eine lokale Variable und initialisieren Sie sie mit dem Zeichenketten-Literal „Hello World“! [2]

c) Prüfen Sie, ob genau ein Kommandozeilenargument übergeben wurde! [2]

d) Wenn ja, so weisen Sie es der lokalen Variablen zu! [2]

e) Wandeln Sie die Zeichenkette um in Kleinbuchstaben! [2]

f) Rufen Sie eine (noch zu erstellende) statische Methode namens „print“ auf, an welche die Zeichenkette als Argument zu übergeben ist! [2]

g) Implementieren Sie diese Methode! [2]

h) Definieren Sie darin drei lokale Variablen, wie folgt: [2]

- `start vom Typ char initialisiert mit dem Zeichen ‘a’`
- `c vom Typ char initialisiert mit dem Zeichen ‘\0'`
- `index vom Typ int initialisiert mit dem Zahlenwert -1`

i) Schachteln Sie zwei Zählschleifen! [2]

j) Lassen Sie die äußere von 0 bis Alphabet.letters[0].length laufen! [2]

k) Lassen Sie die innere über die gesamte Zeichenkette iterieren! [2]

l) Weisen Sie das aktuelle Zeichen der Zeichenkette der Variablen c zu! [2]

m) Bestimmen Sie den Wert der Variablen index als Differenz aus c und start! [2]

n) Prüfen Sie, ob index zwischen 0 und 25 liegt! Falls ja, so geben Sie das aktuelle Element des Arrays „letters“ auf der Konsole aus! [2]

Hinweis:

- `erste Dimension: Wert der Variablen „index“`
- `zweite Dimension: Laufindex der äußeren Schleife`

o) Prüfen Sie anderenfalls, ob c ein Leerzeichen ist! Geben Sie in diesem Falle vier Leerzeichen (als Leerraum zwischen zwei Worten) auf der Konsole aus! [2]

Ergebnis: Das an der Kommandozeile eingegebene Wort wird als ASCII-Art ausgegeben.

Aufgabe 3: Constant [10]

Zweck: Verwenden vordefinierter und Verhindern falscher Ampelfarben bzw. -texte.

a) Erstellen Sie eine Klasse „Ampelfarbe“ mit einem Attribut „text“ vom Typ „String“! Überschreiben Sie die geerbte „toString“-Methode, so dass der Attributwert zurückgegeben wird! [2]

b) Erstellen Sie einen Konstruktor, der einen Initialwert für das Attribut entgegennimmt! Passen Sie seine Sichtbarkeit so an, dass von außen keine Objekte erzeugt werden können! [2]

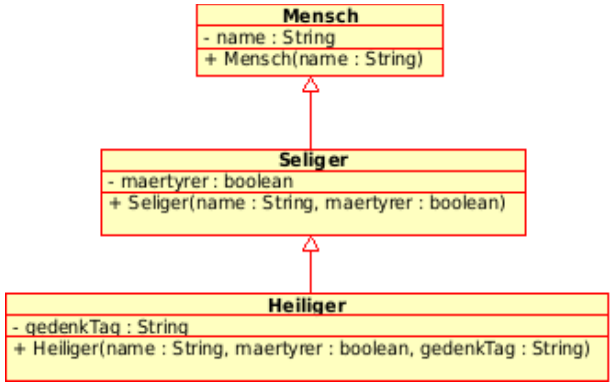
c) Definieren Sie in der Klasse drei Konstanten ROT, GELB, GRUEN des Typs „Ampelfarbe“! [2]

d) Weisen Sie ihnen jeweils ein neu erstelltes Objekt zu, wobei an den Konstruktor als Argument die Worte „Anhalten“, „Achtung“ und „Weiterfahren“ zu übergeben sind! [2]

e) Geben Sie in der „main“-Methode der Startklasse die Werte der drei Konstanten auf der Konsole aus! [2]

Ergebnis: Die Konstanten wurden verwendet und ihre Werte korrekt ausgegeben.

Aufgabe 4: Object [20]



Zweck: Implementieren einer kleinen Software-Architektur und Nutzen der Polymorphie.

Gegeben sei dazu ein Klassendiagramm in Unified Modeling Language (UML)-Notation.

a) Implementieren Sie die Klassen! [2]

b) Berücksichtigen Sie die Vererbung! [2]

c) Fügen Sie die Attribute hinzu! Beachten Sie die Sichtbarkeiten! [2]

d) Fügen Sie die Konstruktoren hinzu! Weisen Sie übergebene Parameterwerte den Attributen zu! [2]

e) Rufen Sie in den Konstruktoren den jeweils passenden Super-Konstruktor auf und geben Sie ihm die geforderten Argumente für die Initialisierung mit! [2]

f) Implementieren Sie die „toString“-Methode in allen drei Klassen, so dass alle Attributwerte zurückgegeben werden! [2]

g) Erzeugen Sie in der „main“-Methode der Startklasse ein Array des Types „Mensch“ mit einer Mindestgröße von drei! [2]

h) Fügen Sie ihm mindestens drei Objekte hinzu: eines vom Typ „Mensch“; ein weiteres vom Typ „Seliger“; ein drittes vom Typ „Heiliger“! [2]

Vorschläge:

Typ	Name	Märtyrer	Gedenktag
Mensch	Hans im Glück	-	-
Seliger	Adolph Kolping	nein	-
Heiliger	Nikolaus von Myra	nein	6. Dezember
Heiliger	Hildegard von Bingen	nein	17. September
Heiliger	Niklaus von Flüe	nein	25. September
Heiliger	Johanna von Orléans	ja	30. Mai
Heiliger	Maximilian Kolbe	ja	14. August

i) Iterieren Sie per for-each-Schleife durch das Array! [2]

j) Geben Sie das jeweilige Array-Element (den Menschen) auf der Konsole aus! [2]

Ergebnis: Das Klassengerüst wurde implementiert und entsprechende Objekte verwendet.

Aufgabe 5: Exception [20]

Zweck: Werfen einer Exception wenn eine Zufallszahl Null ist und Abfangen derselben.

a) Erstellen Sie eine Startklasse mit der Konstanten MAX und dem Ganzzahlwert 500! [2]

b) Erstellen Sie desweiteren eine Klassenmethode namens „getRandomNumber“ mit „int“-Rückgabewert! [2]

c) Generieren Sie darin via „random“-Methode der „Math“-Klasse eine Zufallszahl! [2]

d) Transponieren Sie diese in den Wertebereich zwischen 0 und 99 (inklusive)! [2]

e) Konvertieren Sie den Wert in eine Ganzzahl! [2]

f) Werfen Sie eine Standard-Exception mit kurzer Textbotschaft, wenn der Wert der konvertierten Zufallszahl (arithmetisch) Null ist! [2]

g) Geben Sie anderenfalls die konvertierte Zufallszahl als Ergebniswert zurück! [2]

h) Iterieren Sie in der „main“-Methode per Zählschleife von 0 bis MAX! [2]

i) Rufen Sie die „getRandomNumber“-Methode wiederholt auf und geben ihren Rückgabewert auf der Konsole aus! [2]

j) Fangen Sie eine möglicherweise geworfene Exception ab und geben die Fehlermeldung auf der Konsole aus! [2]

Ergebnis: Es werden 500 Zufallszahlen ausgegeben. Bei jeder Null steht eine Fehlermeldung.

Viel Erfolg!