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 2019-2
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	März 2020
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	10	20	20	20	30	100

Im Rahmen der Klausur sind fünf verschiedene Aufgaben zu lösen.

Aufgabe 1: Array „Gruß“ [10]

Zweck: Erzeugen einer zufälligen Begrüßung.

a) Erstellen Sie eine Startklasse mit Einstiegsmethode! [2]

b) Deklarieren Sie ein Feld (Array), das Zeichenketten enthalten kann! [2]

c) Weisen Sie ihm initial vier Literale zu: "Hallo", "Guten Tag", "Servus", "Moin Moin"! [2]

d) Erzeugen Sie eine Pseudo-Zufallszahl zwischen arithmetisch 0 und 3 (beide inklusive)! [2]

e) Geben Sie das Feldelement auf der Konsole aus, welches am Index liegt, der durch die Pesudo-Zufallszahl bestimmt wird! [2]

Ergebnis: Das Programm gibt eine beliebige Begrüßungsformel auf der Konsole aus.

Aufgabe 2: Operation „Quadrat“ [20]

Zweck: Quadrieren einer Zahl.

a) Definieren Sie eine Zahl als lokale Variable des Typs „long“ mit dem Initialwert 16! [2]

b) Geben Sie ihren Wert auf der Konsole aus! Wandeln Sie den „long“-Wert dazu explizit in eine Zeichenkette um! [2]

c) Fügen Sie eine Zählschleife mit fünf Iterationen ein, deren Laufvariable von 1 bis 5 läuft! [2]

d) Quadrieren Sie die oben definierte Zahl! [2]

e) Geben Sie die einzelnen Zwischenwerte für jeden Schleifendurchlauf auf der Konsole aus! [2]

f) Erstellen Sie ein zweites, vom ersten unabhängiges Programm! [2]

g) Definieren Sie eine Zahl als lokale Variable des Typs „BigInteger“ mit dem Initialwert 16! [2]

h) Geben Sie ihren Wert auf der Konsole aus! Wandeln Sie den „BigInteger“-Wert dazu explizit in eine Zeichenkette um! [2]

i) Nutzen Sie eine zum ersten Programm identische Schleife, um die oben definierte Zahl zu quadrieren! [2]

j) Geben Sie die einzelnen Zwischenwerte für jeden Schleifendurchlauf auf der Konsole aus! [2]

Ergebnis: Die vierte und fünfte Iteration funktioniert nur mit BigInteger.

Aufgabe 3: String „Adresse“ [20]

Zweck: Erstellen und suchen einer Adresse.

a) Erstellen Sie eine Klasse namens „Address“ mit den drei privaten Zeichenkettenattributen „name“, „strasze“, „ort“! [2]

b) Erstellen Sie einen initialisierenden Konstruktor, der alle drei Attribute berücksichtigt! [2]

c) Erstellen Sie öffentliche Zugriffsmethoden für alle drei Attribute! [2]

d) Überschreiben Sie die geerbte „toString“-Methode! Lassen Sie sie eine vernünftig formatierte Ausgabe aller drei Attributwerte zurückliefern! [2]

e) Erzeugen Sie in der neu zu erstellenden Startklasse ein Feld (Array), das fünf „Address“-Objekte speichern kann! [2]

f) Weisen Sie ihm fünf „Address“-Objekte zu! Nutzen Sie dafür den initialisierenden Konstruktor! [2]

g) Erstellen Sie eine statische Methode „print“, der ein „Address“-Feld (Array) als Parameter übergeben wird! Verwenden Sie darin eine „for-each“-Schleife, um alle „Address“-Objekte auf der Konsole auszugeben! [2]

h) Erstellen Sie eine weitere statische Methode „search“, die neben dem „Address“-Feld (Array) auch eine Such-Zeichenkette entgegennimmt! [2]

i) Durchsuchen Sie darin die Namen sämtlicher „Address“-Objekte per Zählschleife! Geben Sie den Index eines gefundenen Objektes als Rückgabewert zurück! [2]

Hinweis: Methode „contains“

j) Testen Sie die Methoden „print“ und „search“ durch Aufruf in der „main“-Methode! [2]

Ergebnis: Beide Methodenaufrufe führen zum erwarteten Ergebnis.

Aufgabe 4: OOP „Vergleich“ [20]

Zweck: Vergleich zweier Metallplatten.

a) Erstellen Sie eine Klasse namens „MetallPlatte“ mit zwei Gleitkommazahlen als Attribute „laenge“ und „breite“! [2]

b) Erstellen Sie einen initialisierenden Konstruktor, der für beide Attribute jeweils ein Argument entgegennimmt! [2]

c) Geben Sie der Klasse eine Methode „berechneFlaeche“, die beide Attribute als Operanden verwendet und den Ergebniswert vom Typ „double“ zurückliefert! [2]

d) Geben Sie der Klasse eine weitere Instanzmethode namens „vergleicheMit“, die ein Argument vom Typ „MetallPlatte“ entgegennimmt! [2]

e) Vergleichen Sie darin die Fläche des aktuellen mit jener des übergebenen Objektes! [2]

f) Geben Sie als Ergebnis eine der beiden Zeichenketten „größer als“ oder „kleiner als oder gleich zu“ zurück! [2]

g) Verwenden Sie den ternären Operator für obigen Vergleich! [2]

h) Erstellen Sie eine Startklasse mit „main“-Methode! Erzeugen Sie darin drei Objekte des Typs „MetallPlatte“, von denen zwei gleich sind und die dritte kleinere Maße hat! [2]

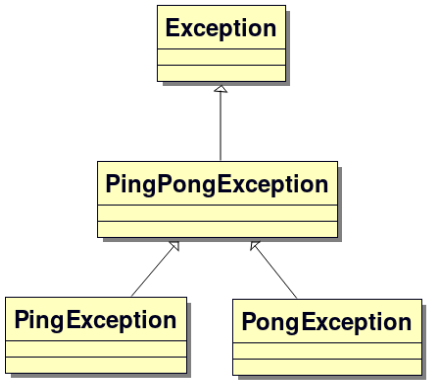
i) Vergleichen Sie unter Verwendung der Methode „vergleicheMit“ jeweils Platte 1 und 2 sowie anschließend Platte 1 und 3! Speichern Sie die Rückgabewerte in zwei lokalen Variablen! [2]

j) Geben Sie die beiden Variablen als Ergebnis auf der Konsole aus! [2]

Ergebnis: Die Vergleiche werden auf Basis der berechneten Flächen korrekt durchgeführt.

Aufgabe 5: Exception „Menüauswahl“ [30]

Zweck: Abfangen von Ausnahmen in einer Anwendung.



a) Erstellen Sie die Klasse „PingPongException“ gemäß gegebenem Klassendiagramm! [2]

b) Erstellen Sie außerdem die beiden Klassen „PingException“ und „PongException“! [2]

c) Erstellen Sie die übliche Startklasse namens „Launcher“ mit „main“-Methode! [2]

d) Erstellen Sie dort drei weitere statische Methoden namens „pingPong“, „ping“ und „pong“! Lassen Sie sie einen kurzen Text wie zum Beispiel „Methode ‘pingPong’ wurde aufgerufen.“ auf der Konsole ausgeben! [2]

e) Passen Sie ihre Signaturen so an, dass jeweils eine namentlich passende, zu werfende Ausnahme (PingPongException, PingException, PongException) deklariert wird! [2]

f) Werfen Sie in den drei Methoden jeweils eine namentlich passende Ausnahme (PingPongException, PingException, PongException)! [2]

g) Implementieren Sie nunmehr die Klassenmethode „showMenu“! Berücksichtigen Sie in ihrer Signatur, dass alle drei Ausnahmearten (PingPongException, PingException, PongException) geworfen werden können! [2]

h) Geben Sie zunächst auf der Konsole ein kleines Menü mit folgenden Optionen aus: 1 – Ping; 2 – Pong; 3 – PingPong! Bitten Sie den Anwender dann um Eingabe seiner Wahl! [2]

i) Nehmen Sie mittels „Scanner“-Klasse seine Auswahl als Ganzzahl entgegen und speichern Sie sie in einer lokalen Variablen! [2]

j) Geben Sie das „Scanner“-Objekt als Ressource wieder frei! [2]

k) Unterscheiden Sie je nach erfolgter Auswahl mittels „switch-case“-Konstrukt! [2]

l) Rufen Sie eine passende statische Methode auf! [2]

m) Melden Sie im „default“-Zweig per kurzer Textausgabe einen Fehler über die Fehlerkonsole (NICHT Standard-Konsole)! [2]

n) Rufen Sie in der „main“-Methode die statische Methode „showMenu“ auf! [2]

o) Fangen Sie alle drei möglichen Ausnahmetypen ab! [2]

Ergebnis: Alle drei Ausnahmearten können erfolgreich provoziert werden.

Viel Erfolg!