Examination
Data Processing (DP)

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

Personal Data	
First and Last Name	Ihre Daten werden von der Klausuraufsichtsperson schriftlich auf der Anwesenheitsliste festgehalten.
Matriculation Number	
Subject and Year	CS 2016-1
Login	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!

Examination Data	
Date	2017-06-23
Duration [min]	120
Maximum Points [Point]	100
Permitted Study Aids	Dokumentation im lokalen Netzwerkverzeichnis (Intranet); NICHT gestattet sind Kommunikationsmöglichkeiten (Internet) oder Anmeldung via SSH auf dem Rechner "fileserv", wo Ihr Homeverzeichnis liegt, oder eine Anmeldung mit Ihrem Klausur-Login nach Ende der Prüfung. Dies kann leicht geprüft werden (last cs12*, Server-Log-Dateien). Bitte unterlassen Sie also Täuschungsversuche in Ihrem eigenen Interesse.
Remarks	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. Diese Aufgabenstellung in Papierform können Sie nach dem Ende der Klausur behalten.

Evaluation											
Task	1	2	3	4	5	6	7	8	9	10	Summe
Points	20	20	30	20	10	0	0	0	0	0	100

Die Nutzung der Java-Klassenbibliothek wird innerhalb fünfer Aufgaben geprüft. Dabei liegt der Fokus auf der Verarbeitung von Daten verschiedenster Formate.

Task 1: Container [20]

Zweck: Zählen der Häufigkeit von Wörtern in einem Text.

Hintergrundinformation: Zum schnellen Zugriff auf die in ihm gespeicherten Werte wird ein so genannter "Rot-Schwarz-Baum" (RB tree) vom Typ Binärer Suchbaum verwendet.

a) Erzeugen Sie eine Dateireferenz auf die gegebene Datei "gpl.txt"! [2]

b) Nutzen Sie diese zum Erzeugen eines Objektes der Klasse "Scanner"! [2]

c) Instanziieren Sie die Datenstruktur "TreeMap", wobei die Schlüssel Zeichenketten und die Werte Ganzzahlen sein sollen! [2]

d) Iterieren Sie mittels "while"-Schleife und "Scanner"-Objekt Wort für Wort durch den Dateiinhalt! [2]

e) Speichern Sie das jeweils aktuelle Wort in einer lokalen Variablen! [2]

f) Prüfen Sie, ob das Wort im Container bereits vorhanden ist! [2]

g) Fügen Sie das Wort als Schlüssel zum Container hinzu und geben Sie als zugehörigen Wert die Häufigkeit des Wortes an! [2]

Hinweis: Enthält der Container das Wort bereits, so ist seine Häufigkeit um Eins zu erhöhen.

h) Schließen Sie den Eingabedatenstrom! [2]

i) Verwenden Sie, nachdem der komplette Dateiinhalt ausgelesen wurde, eine "for-each"-Schleife, um den Containerinhalt (Schlüssel und zugehörige Werte) auf der Konsole auszugeben! [2]

j) Geben Sie außerdem die Größe des Containers (Anzahl der enthaltenen Elemente) aus! [2]

Zusatzaufgabe k) Schränken Sie die einzulesenden Zeichen im "Scanner"-Objekt durch Angabe des folgenden Musters ein [2]:

```
"[^a-zA-Z']+"
```

Ergebnis: Alle in der Datei enthaltenen Wörter wurden mit ihrer Häufigkeit auf der Konsole ausgegeben.

Task 2: Stream [20]

Zweck: Der Quelltext einer Webseite soll ausgelesen werden.

a) Erzeugen Sie eine Instanz der Klasse "URL", welche auf http://www.ba-leipzig.de/ verweist! [2]

b) Öffnen Sie mittels "openConnection" eine Verbindung und speichern Sie sie in einer lokalen Variablen! [2]

c) Bestimmen Sie aus ihr den Eingabedatenstrom! [2]

d) Erzeugen Sie mittels Eingabedatenstrom ein "InputStreamReader"-Objekt! [2]

e) Übergeben Sie dieses schließlich an den Konstruktor der Klasse "Scanner"! [2]

f) Verwenden Sie eine "while"-Schleife zum fragmentweisen Auslesen des kompletten Eingabedatenstromes mittels "Scanner"-Objekt! [2]

g) Geben Sie alle eingelesenen Zeichen auf der Konsole aus! [2]

h) Nutzen Sie eine Zählvariable zum Mitzählen der eingelesenen Fragmente (Tokens)! Geben Sie ihren Wert nach Verlassen der Schleife aus! [2]

i) Schließen Sie den oder die geöffneten Systemressourcen (Stream)! [2]

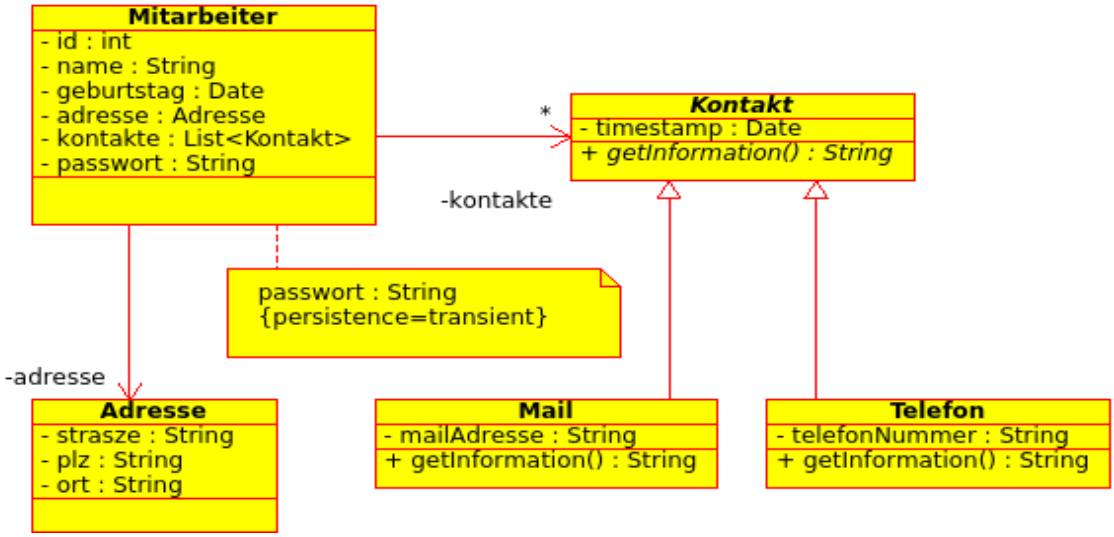
Hinweis: Ein Methodenaufruf reicht. Wichtig ist, an welchem Objekt.

j) Fangen Sie eventuell auftretende Ausnahmen ab! [2]

Ergebnis: Die Webseite wurde auf Konsole ausgegeben und ihre Zeichen gezählt.

Task 3: Java Object Serialisation (JOS) [30]

Zweck: Java-Objekte sollen mittels JOS serialisiert und später rekonstruiert werden. Gegeben Sei dazu ein UML-Klassendiagramm. Ein Mitarbeiter hat eine Adresse und gegebenenfalls mehrere Kontaktinformationen (Mail-Adressen, Telefonnummern).



a) Implementieren Sie die im Diagramm gezeigten Klassen in Java-Quelltext! [2]

b) Machen Sie die Klasse "Kontakt" sowie ihre Methode "getInformation" abstrakt! [2]

c) Berücksichtigen Sie die Vererbung! [2]

d) Zeigen Sie für alle fünf Klassen mittels Schnittstelle an, dass sie serialisierbar sind! [2]

Hinweis: Ignorieren Sie Warnungen des Compilers bezüglich "serialVersionUID"! [2]

e) Fügen Sie Attribute bzw. Assoziationen hinzu! [2]

f) Kennzeichnen Sie das Attribut "passwort" der Klasse "Mitarbeiter" als flüchtig! [2]

g) Erstellen Sie Zugriffsmethoden für alle Attribute aller Klassen! [2]

h) Deklarieren bzw. Implementieren Sie weitere gegebene Methoden! [2]

i) Beachten Sie die Sichtbarkeiten! [2]

j) Schreiben Sie ein Programm, das zwei Objekte vom Typ "Mitarbeiter" erzeugt und mit beliebigen Werten initialisiert! Weisen Sie neben primitiven Werten auch zusammengesetzte Objekte vom Typ "Address" und "Kontakt" zu! [2]

k) Erzeugen Sie je eine Instanz der Klassen "ObjectOutputStream" und "XMLEncoder" und übergeben ihren Konstruktoren jeweils ein passendes Objekt vom Typ "FileOutputStream"! [2]

l) Serialisieren Sie beide oben erzeugten Objekte vom Typ "Mitarbeiter", einmal als Byte-Strom und einmal im XML-Format! [2]

m) Erstellen Sie ein zweites, unabhängiges Programm (Startklasse mit "main"-Methode)! Lesen Sie darin die beiden serialisierten Objekte wieder ein! [2]

n) Flexibilisieren Sie das Programm durch Verwendung einer Schleife zum Einlesen einer unbekannten Zahl von Objekten! [2]

Hinweis: Das Ende des Datenstromes kann erkannt werden durch Abfangen einer Exception:

- XMLDecoder: ArrayIndexOutOfBoundsException
- ObjectInputStream: EOFException

o) Geben Sie ihre Eigenschaften (bzw. Attributwerte) auf der Konsole aus! [2]

Ergebnis: Beide Objekte wurden korrekt eingelesen und ihre Daten ausgegeben.

Task 4: Comparison [20]

Zweck: Ein Spielwürfel soll simuliert werden.

a) Erstellen Sie eine Klasse namens "Die" mit den zwei Instanz-Attributen "sides" und "result", jeweils als ganze Zahl! [2]

b) Geben Sie der Klasse eine Integer-Konstante namens "DEFAULT_SIDES" mit dem Wert 6! [2]

c) Fügen Sie ihr schließlich noch eine Klassenvariable namens "generator" vom Typ "Random" hinzu, deren Objekt direkt bei der Deklaration (nicht im Konstruktor) zu erstellen ist! [2]

d) Implementieren Sie einen Konstruktor, der die zwei Parameter "sides" und "result" entgegennimmt, welche dem neuen Würfelobjekt zugeordnet werden! Implementieren Sie einen zweiten Konstruktor, der nur die Seitenanzahl entgegennimmt und als Standardseite die Eins festlegt! [2]

e) Fügen Sie als dritten den Standard-Konstruktor hinzu! Intern soll keine direkte Zuweisung von Werten erfolgen, sondern der zuvor erstellte (zweite) Konstruktor mit der Konstanten "DEFAULT_SIDES" aufgerufen werden. [2]

f) Erstellen Sie eine Methode "roll", um einen neuen Wert würfeln zu können! Nutzen Sie dazu das Attribut "generator"! Weisen Sie das Ergebnis dem Attribut "result" zu! [2]

g) Implementieren Sie die geerbte Methode "equals" in passender Weise! [2]

h) Implementieren Sie die geerbte Methode "toString" in passender Weise! [2]

i) Erzeugen Sie in der "main"-Methode drei Würfel: die ersten beiden mittels Standard-Konstruktor und den dritten unter Angabe der Seitenanzahl 10! Würfeln Sie alle drei Würfel je einmal! [2]

j) Vergleichen Sie den ersten und zweiten Würfel mittels "equals"-Methode und geben das Ergebnis auf der Konsole aus! Geben Sie den dritten Würfel mittels "toString"-Methode auf der Konsole aus! [2]

k Zusatz) Nutzen Sie im Konstruktor mit den zwei Parametern "sides" und "result" das Schlüsselwort "assert", um zu prüfen, ob die Seitenanzahl größer als Eins ist und ob sich die Punktzahl des Würfels innerhalb des gültigen Wertebereiches befindet! [2]

Ergebnis: Die Methoden des Würfels wurden erfolgreich angewandt.

Task 5: Date and Time [10]

Zweck: Einige Datumsangaben sollen auf der Konsole ausgegeben werden.

a) Nutzen Sie in einer Startklasse mit "main"-Methode eine Instanz der Klasse "Calendar"! [2]

b) Bestimmen Sie die aktuelle Woche im Monat und geben Sie sie auf der Konsole aus! [2]

c) Bestimmen Sie die aktuelle Kalenderwoche (KW) und geben Sie sie auf der Konsole aus! [2]

d) Addieren Sie eine Woche zum Kalenderfeld "WEEK_OF_MONTH" hinzu! [2]

e) Geben Sie das in der Kalenderinstanz eingestellte Datum im Format Tag-Monat-Jahr aus! [2]

Ergebnis: Die Kalenderklasse wurde erfolgreich angewandt.

Viel Erfolg!