Name:

Geburtsdatum:

Matrikelnummer:

Klausur
Informatische Werkzeuge in den Geistes- und Sozialwissenschaften 2

25. Juli 2019

	Nur zur Korrektur, bitte freilassen!									
Aufgabe	1.1	1.2	2.1	3.1	3.2	4.1	5.1	6.1	Summe	Note
Möglich	4	5	20	3	8	10	4	6	60	
Erreicht										

Klausurnote: Bonuspunkte: Endnote:

The „Lösungen“ der Aufgaben in diesem Dokument sollen den Studierenden als Anfangspunkt für die Beantwortung der Aufgaben dienen. Trotz aller Bemühungen kann es zu unvollständigen oder sogar Fehlern in den „Lösungen“ kommen. Da die Korrektur und Benotung der gestellten Aufgaben niemals lediglich auf einen „Vergleich mit der Musterlösung“ hinausläuft, ist dies auch nicht so schlimm. In jedem Fall sollten die Studierenden die Lösungen nachvollziehen und im Prinzip mittels des Lehrstoffs selbst verifizieren können.

Sollten Sie „Lösungen“ finden, die Sie nicht verstehen oder sogar für fehlerhaft halten diskutieren Sie diese am besten mit den Tutoren oder auf dem Kursforum und benachrichtigen Sie die Lehrenden.

1 Versionskontrolle

Aufgabe 1.1 Bera Beispielperson führt in ihrer Shell folgende Befehle aus, um ein `git`-Repository zu klonen: 4 Pkt 4 min

```
git clone https://gitlab.cs.fau.de/iwgs-ss19/collaboration.git
cd collaboration
```

Sie verändert danach lokal die Datei "users.txt", indem sie einen Eintrag für sich selbst hinzufügt und führt folgende Befehle aus, um ihre lokalen Änderungen zu veröffentlichen:

```
git add users.txt
git commit -m "Eintrag zu users.txt hinzugefuegt"
git pull
git push
```

Beschreiben Sie, warum Bera vor `git push` noch ein `git pull` ausgeführt hat und welches Problem auftreten könnte, wenn sie diesen Schritt überspringen würde.

Lösung:In der Zeit, in der Bera ihre Änderungen durchgeführt hat, hätte jemand anderes Änderungen veröffentlichen können. Ist dies passiert, so wird Beras `push` abgelehnt werden.

Aufgabe 1.2 Erklären Sie das Konzept eines *merge conflicts*, unter welchen Umständen dieses Problem auftreten kann und wie es von einem Entwickelnden behoben werden kann. 5 Pkt 5 min

Lösung:Ein *merge conflict* tritt auf, wenn zwei Entwickelnde unabhängig die gleiche Stelle einer Datei editiert haben und es in der Folge nicht möglich ist, beide Änderungen (konsistent) zu veröffentlichen. Der Name *merge conflict* kommt daher, dass dieses Problem auftritt, wenn die einen Änderungen mit den anderen *gemerged* (vereinigt) werden sollen.

Entwickelnde können einen merge conflict beheben indem sie manuell auswählen, welche der rivalisierenden Änderungen angenommen werden soll.

2 Datenbanken

Aufgabe 2.1 (Mitarbeitendenverwaltung)
Gegeben ist folgender Datensatz: 20 Pkt 20 min

PersonalID	Vorname	Nachname	Projekt	Projektstart
123	Erika	Musterfrau	Backend	2009
137	Jan	Modaal	Bildverarbeitung	2015
055	Ola	Nordmann	Bildverarbeitung	2015
249	Juan	Perez	Frontend	2010
005	Zhang	San	Backend	2009

Die PersonalID identifiziert einen Datensatz eindeutig (Primary Key). Sie wissen außerdem, dass für einen gleichen Eintrag in der Spalte **Projekt** auch der Eintrag in der Spalte **Projektstart** gleich sein wird.

1. **Tabelle Erstellen (5 Punkte)** Geben Sie die SQL-Befehle zum Erstellen einer Tabelle unter dem Namen Employees mit der obigen Struktur an. Sie müssen keine Einträge hinzufügen. Vermeiden Sie dabei eventuelle Probleme, falls bereits eine Tabelle namens Employees existiert.

2. **Daten anpassen (4 Punkte)**: Ola Nordmann wechselt von ihrem bisherigen Projekt (Bildverarbeitung) in das Backend-Projekt. Geben Sie den SQL-Ausdruck an, der den entsprechenden Datensatz anpasst.

3. **Redundanz (3 Punkte)**: In der obigen Tabelle werden einige Daten mehrfach gespeichert. Begründen Sie, warum das (gerade für größere Datensätze) suboptimal ist.

4. **Praxis (8 Punkte)**: Die obige Tabelle wurde in einer SQL-Datenbank database.db in einer Tabelle Employees gespeichert. Schreiben Sie ein python-Programm, welches diese Datenbank-Tabelle in ein Textdokument mit dem Namen employees.txt schreibt. Formatieren Sie das Dokument so, dass jeder Datensatz in eine neue Zeile geschrieben wird und die einzelnen Felder durch Kommas getrennt sind.

Hinweis: Mit einem File-Objekt fl können Sie die Methode fl.writelines(strings) verwenden, um eine Liste von Strings als Zeilen in eine Datei zu schreiben.

Lösung:

1. Die Befehle sind wie folgt:

```
DROP TABLE IF EXISTS Employees;
CREATE TABLE Images(
    PersonalID INTEGER PRIMARY KEY,
    Vorname TEXT,
    Nachname TEXT,
    Projekt TEXT,
    Projektstart INTEGER);
```

2. Der Befehl lautet wie folgt:

```
UPDATE Employees
SET Projekt = "Backend", Projektstart = "2009"
WHERE PersonalID = 55;
```

3. In der Tabelle wird sowohl der Projektname als auch das Jahr des Projektstarts festgehalten, obwohl bekannt ist, dass diese immer korrelieren. Diese Informationen könnten also (z.B. in eine zweite Tabelle "Projects") ausgelagert werden.

   Die doppelte Vorhaltung der Daten ist problematisch weil so leichter Fehler / Inkonsistenzen entstehen und mehr Speicherplatz verbraucht wird, als nötig.

4. Hier ist ein solches Programm:

```python
import sqlite3

# Start with empty list for entries
entries = []

# Connect to DB and get cursor
db = sqlite3.connect("database.db")
cursor = db.cursor()

# Select the whole database.
cursor.execute("SELECT * FROM Employees")
results = cursor.fetchall()

# Loop over all entries in the database.
for ent in results:

  # For every ent(ry in the database) we build a new entry
  # for the file we want to print (in form of a string).
  entry = ""
  for value in ent:
    # Not all values are strings so we have to cast.
    entry = entry + str(value) + ","

  # The code above produces a trailing comma so we don't
  # append the whole thing, just everything before that.
  entries.append(entry[0:-1])

# Write all entries to file.
with open("employees.txt", "w") as fl:
  fl.writelines(entries)

# Close the database.
db.close()
```

---

3 Bild

Aufgabe 3.1 (Kantenerkennung)

Beschreiben Sie kurz in eigenen Worten, wie Kantenerkennung auf Bildern funktioniert. 3 Pkt

Lösung:In einer Rastergrafik können Kanten erkannt werden, indem der Farbwert eines Pixels mit den Farbwerten der Pixel in seiner *Nachbarschaft* (also die angrenzenden Pixel in alle Richtungen) verglichen wird. 3 min

Ist der Wert gleich oder unter einem gewissen (arbiträren) Schwellenwert, so liegt keine Kante vor. Ist der Wert stark unterschiedlich, wurde möglicherweise eine Kante entdeckt.

Anfang und Ende der Kante können gefunden werden, wenn man ganze Nachbarschaften von Pixeln vergleicht und sieht, wo Unterschiede anfangen und enden.

.

Aufgabe 3.2 (Alpha-Blending)

Schreiben Sie eine python-Funktion alphaBlend(bottomImage, topImage). Die beiden Argumente sind Bilder, repräsentiert als Listen von Listen (von Pixeln). 8 Pkt 8 min

- Die Einträge der inneren Listen sind alle jeweils ein Pixel, hier ein Tupel im Format (R,G,B,A).
- Die Pixelwerte in jedem Kanal sind normalisiert, also im Bereich zwischen 0 und 1.
- Ihre Funktion soll topImage über bottomImage legen, unter Beachtung des Alpha-Kanals.

Sie können für diese Aufgabe annehmen, dass topImage und bottomImage die gleichen Dimensionen haben.

Lösung:Hier ist eine mögliche Implementation. Zu beachten ist, dass wir hier davon ausgehen, dass bottomImage komplett opak ist, wie im Beispiel in der Vorlesung.

```
def alphaBlend(bottomImage, topImage):
    # we know the images have the same dimension
    xlen = len(top)
    ylen = len(top[0])

    # construct new picture on the go.
    pic = []

    for x in range(xlen):
        row = []
        for y in range(ylen):
            bg = bottomImage[x][y] # background
            fg = topImage[x][y]   # foreground

            # Only alpha from top image is relevant
            alpha = fg[3]

            # as explained on Slide 253
            px_red = ((1-alpha) * bg[0]) + (alpha * fg[0])
            px_gre = ((1-alpha) * bg[1]) + (alpha * fg[1])
            px_blu = ((1-alpha) * bg[2]) + (alpha * fg[2])

            # create and append new pixel to row
            px = (px_red, px_gre, px_blu, 1.0)
            row.append(px)

        # append completed row to picture
        pic.append(row)

    return pic
```

4 Web-Applikationen

Aufgabe 4.1 (Web-Applikation mit Liste)

Gegeben ist ein Bottle-Webserver. Dieser besteht aus zwei Dateien Server.py und List.tpl. 10 Pkt

Vervollständigen Sie folgendes Code-Gerüst, sodass auf der Webseite eine Überschrift und eine Liste angezeigt werden. Beide sollen in Abhängigkeit von den übergebenen Daten generiert werden: In der Überschrift soll der Wert der übergebenen Variable heading stehen und die Liste soll aus den Werten in listData bestehen. 10 min

```
# DATEI: Server.py

@bottle.route('/list')
def showList() :
    names = ['Michael', 'Jonas', 'Philipp']
    return bottle.template('List.tpl', heading='My list', listData=names)
```

```
<!-- Datei: List.tpl -->
<html>

  <head>



  </head>

  <body>

  </body>
</html>
```

Lösung:Here's the full template:

```
<html>
    <head>
        <title>{{heading}}</title>
```

```
    </head>
    <body>
        Here's a list:
        <ul>
            <%
            for element in listData:
            %>
                <li>{{element}}</li>
            % end
        </ul>
    </body>
</html>
```

5 Geistiges Eigentum

Aufgabe 5.1 (CopyLeft)

Nennen Sie die „Copyleft"-Klausel in der GNU Public License oder in den Creative Commons Lizenzen, und erklären Sie kurz in eigenen Worten die Wirkungsweise. 4 Pkt 4 min

Lösung:Die „CopyLeft"-Klausel besagt, dass wen ein vom lizenzierten Werk abgeleitetes Werk weiterverbreitet wird, dann muss es mit genau der gleichen Lizenz lizensiert werden wie das Original.

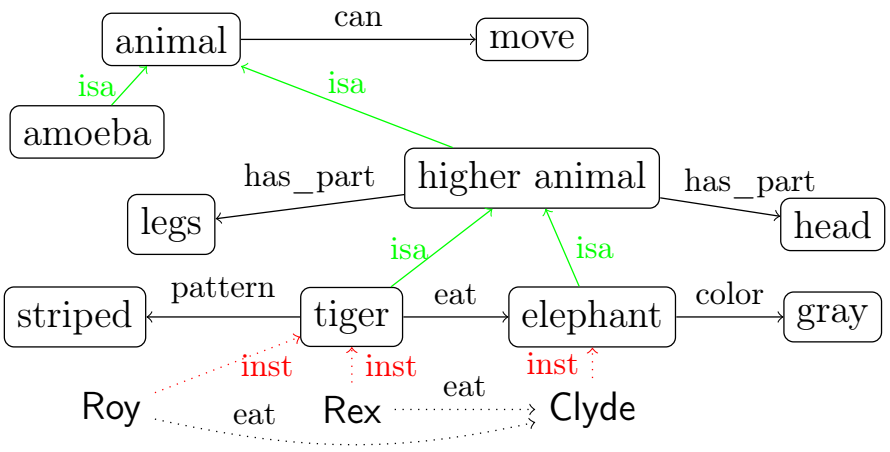
Dadurch wird sichergestellt, dass alle, die auf ein so lizensiertes Werk aufbauen, sich entscheiden müssen, ob sie

- es weiterverbreiten wollen – dann müß das abgeleitete Werk dem „CopyLeft" unterliegen und so dem gemeinfreien Bereich zugeordnet werden, oder
- nicht, dann muss das abgeleitete Werk nicht lizensiert werden, (aber die Schöpfer bekommen erzielen keinen Mehrwert aus Verbreitung oder Verkauf).

6 Semantic Web

Aufgabe 6.1 (Semantische Netze)

Gegeben sei das folgende semantische Netz: 6 Pkt 6 min



1. Geben Sie je drei Subjekt/Prädikat/Objekt-Tripel aus der TBox und der ABox dieses Netzes an.

2. Zeichnen Sie jeweils drei Fakten und terminologische Axiome in das Netz ein, die im Netz inferiert werden können aber nicht explizit repräsentiert sind.

Lösung:

1. Subjekt/Prädikat/Objekt

TBox	ABox
animal/can/move	Rex/inst/tiger
tiger/pattern/striped	Rex/eat/Clyde
amoeba/isa/animal	Roy/eat/Clyde

2. Wir zeichnen die neuen Relationen in blau ein: