Die Hausaufgaben müssen von jedem Studierenden einzeln bearbeitet und abgegeben werden. Für die Hausaufgabe sind die aktuellen Informationen vom Blog `https://seblog.cs.uni-kassel.de/ws1920/programming-methodologies/` zu berücksichtigen.

Abgabefrist ist der 31.10.2019 - 23:59 Uhr

Vorbereitung

Für die Abgabe der Hausaufgaben wird ein Git-Repository genutzt. Dieses wird von jedem Studierenden selbst angelegt und ist ausschließlich für den jeweiligen Studierenden sowie für die Betreuer und Korrekteure sichtbar.

Nicht, oder zu spät gepushte (Teil-)Abgaben werden mit 0 Punkten bewertet.

Zunächst muss ein Account auf `https://github.com/` angelegt werden (es kann auch ein bereits vorhandener Account genutzt werden). Da Git auch in der späteren Arbeitswelt ein essentielles Tool darstellt, nutzt diese Anmeldung für euch und seht von einem Spaß-Account mit Nutzernamen wie: ”XxXEdg3L0rdXxX” ab.

Ohne gültigen Account kann der folgende Schritt nicht durchgeführt werden.

Meldet euch nun unter `https://classroom.github.com/a/34QjsLuZ` für diese Hausaufgabe an. Folgt den dort geschilderten Anweisung um Zugriff auf das erstellte Repository zu erhalten. Eure Abgaben ladet ihr, wie in der Übung gezeigt, darin hoch. Es werden nur Text(.txt), Markdown(.md) oder PDF(.pdf) als Dateiformat in den jeweiligen Aufgaben verlangt. Andere Formate werden nicht akzeptiert.

Ein Abweichen von diesem Schema führt zu einer Nichtbewertung.

Aufgabe 1 - Git (30P)

In dieser Aufgabe sollen die Grundkonzepte ”Commit”, ”Branch”, ”Checkout” und ”Merge” des Git -Workflows kennengelernt werden. Im Verlauf der Aufgabe wird ein Mergekonflikt hervorgerufen, welcher aufgelöst werden soll. Führe die folgenden Schritte auf dem, in der Vorbereitung erstellten Repository durch.

1. Erstelle einen Ordner mit dem Namen ”taskOne”. Erstelle innerhalb dieses Ordners eine Datei mit dem Namen ”me.txt”. Committe und pushe diese Änderung am Dateisystem
2. Erstelle einen Branch mit dem Namen ”dev”
3. Führe danach einen Check Out auf den ”master”-Branch durch
4. Schreibe den folgenden Satz in die Datei:
   ”Hallo mein Name ist <Emailadresse><Matrikelnummer><Nachname>”. Committe und pushe die Änderung
5. Führe einen Check Out auf den ”dev”-Branch durch
6. Schreib folgenden Satz in die (nun wieder leere) Datei ”me.txt”: ”Hallo mein Name ist 4242Albert”. Committe und pushe anschließend diese Änderung
7. Merge nun den ”dev”-Branch in den ”master”-Branch (dies sollte einen Konflikt hervorrufen).
8. Committe die Änderung mit dem vorhandenen Konflikt in der Datei (Dies ist normalerweise nicht üblich und ein No-go, aber es verdeutlicht in diesem Fall das Problem)
9. Behebe nun den Mergekonflikt. In der Datei soll am Ende wieder folgender Satz stehen: ”Hallo mein Name ist <Matrikelnummer><Nachname>”. Committe und pushe abschließend diese Änderung

Nach Bearbeitung der Aufgabe sollte das Dateisystem des Repositorys wie folgt aussehen.

```
assignment-1-<GitHub-Username>/
├── .git/...
└── taskOne/
    └── me.txt
```

Bei der Bewertung wird vor allem darauf geachtet, dass der Mergekonflikt unaufgelöst committet wurde und dessen Lösung erst im Nachhinein geschieht.

Zur Bearbeitung darf ein beliebiges Git-Tool genutzt werden

Aufgabe 2 - Abstrakt vs. Konkret (30P)

Diese Aufgabe beschäftigt sich mit dem Unterschied zwischen abstrakten und konkreten Bezeichnungen sowie den Begriffen ”Abstrakt” und ”Konkret”. Hierzu wird sich mit dem in Abbildung 1 zu sehenden Wimmelbild auseinandergesetzt. Sollte das Bild zu klein sein, kann die Originaldatei aus der Quelle genutzt werden.

Quelle: `http://www.carmen-hochmann.de/Wimmelbuecher/wimmel-melle-landleben.jpg`

Abbildung 1: Fluss im Herbst

1. Erstelle eine Tabelle mit den Spalten „Abstrakt” und „Konkret”. Trage nun 5 Beispielpaare in die Tabelle ein. Alle Paare müssen in Abbildung 1 zu sehen sein.
2. Definiere auf der Grundlage aus Aufgabenteil 1 die Begriffe „Abstrakt” und „Konkret”.
3. Definiere den Begriff „Beispiel” und stelle hierbei einen Bezug zu den Definitionen aus Aufgabenteil 2 her.

Lege die erstellte/n Datei/en in einem neuen Ordner mit dem Namen ”taskTwo” in deinem Repository ab. Committe und pushe die Änderungen abschließend auf den master-Branch.

Aufgabe 3 - Textuelle Szenarien (40P)

Erstelle **drei** textuelle Szenarien zu konkreten Spielsituationen des Spiels „Shroom Wars“. Die Szenarien sollen in Englisch verfasst sein. Die Regeln des Spiels findest du auf der nächsten Seite vor dem Anhang.

Hinweis: Ein textuelles Szenario besteht IMMER aus einem Titel, einer Startsituation, einer Aktion sowie einer Endsituation. Diese 4 Teile sollten sichtbar (durch Farbe und/oder Absatz) voneinander getrennt sein

Als Hilfestellung liegt ein einfaches Beispielszenario vor, das **nicht** für die Hausaufgabe verwendet/kopiert werden darf:

Title: Target enemy house

Start:	Alice and Bob are playing ”Shroom Wars” . Alice has 4 shrooms at her house. Bob has 2 shrooms at his house.
Action:	Alice sets the target of her house on the house of Bob by clicking twice.
Result:	2 of Alice’s shrooms target the chosen house and start moving.

Lege die erstellte/n Datei/en in einem neuen Ordner mit dem Namen ”taskThree”in deinem Repository ab. Committe und pushe die Änderungen abschließend auf den master-Branch.

Spielregeln

This semester we organize an epic battle of nature. One player and up to 3 com's will fight until their last breath.

Each challenger has a starting house with some shrooms at his house. The battlefield contains multiple, unconquered houses. The goal of the game is to be the last one standing until the other challengers don't have any houses or shrooms left.

The game is organized by a game-loop we call "ticks". The following actions are triggered automatically on every tick:

1. A house produces a certain amount of shrooms
2. A house is conquered by a players shrooms if they decrease a house's HP to 0
3. Shrooms move closer to their set targets.
4. The set target of a house decides what the shrooms at this house do (move to other house or rest)
5. Shrooms that already reached their target fight other shrooms if there are any
6. Shrooms that already reached their target damage the house if there are no shrooms present

The Player can trigger the following actions at any time, while a com can only trigger them at a tick:

1. Set an enemy's house as the target of their own house by clicking:
   - 1x to command 25% of the resting shrooms to move at the next tick
   - 2x to command 50% of the resting shrooms to move at the next tick
   - 3x to command 75% of the resting shrooms to move at the next tick
   - 4x to command 100% of the resting shrooms to move at the next tick
2. Upgrade a house by sacrificing a certain amount of resting shrooms to get a:
   - Tank Garage, that produces "tanky shrooms" that have more HP
   - Magic Hut, that produces "magic shrooms" that have more ATK

Anhang

Es folgt eine Auflistung hilfreicher Webseiten und weiteren Erklärungen zu den Themen dieser Hausaufgabe. Die Links sind als Startpunkt zur selbstständigen Recherche gedacht. Das Durcharbeiten der folgenden Quellen ist kein bewerteter Anteil der Hausaufgaben.

Git

- Download: `https://git-scm.com/downloads`
  - Hilfestellung für Linux: `https://git-scm.com/book/en/v2/Getting-Started-Installing-Git`
  - Hilfestellung für Mac: `https://gist.github.com/derhuerst/1b15ff4652a867391f03#file-mac-md`
- Documentation: `https://book.git-scm.com/doc`
  - What is Version Control?: `https://book.git-scm.com/video/what-is-version-control`
  - Pro Git: `https://book.git-scm.com/book/en/v2`
- A Visual Git Reference: `http://marklodato.github.io/visual-git-guide/index-en.html`
- Git Cheat Sheet: `https://github.github.com/training-kit/downloads/github-git-cheat-sheet.pdf`

GitHub

- GitHub: `https://github.com/`
- Getting started with GitHub `https://help.github.com/en/categories/getting-started-with-github`
- Beispiel für ein GitHub-Repository, schaut euch z. B. Dateien, Commits, Branches, Issues und Pull requests an: `https://github.com/ytdl-org/youtube-dl`