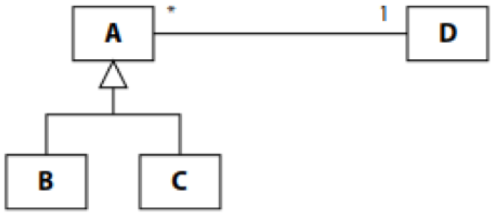
Übungsblatt 05

In diesem Praktikum geht es um GIT, Klassen- und Zustandsdiagramme

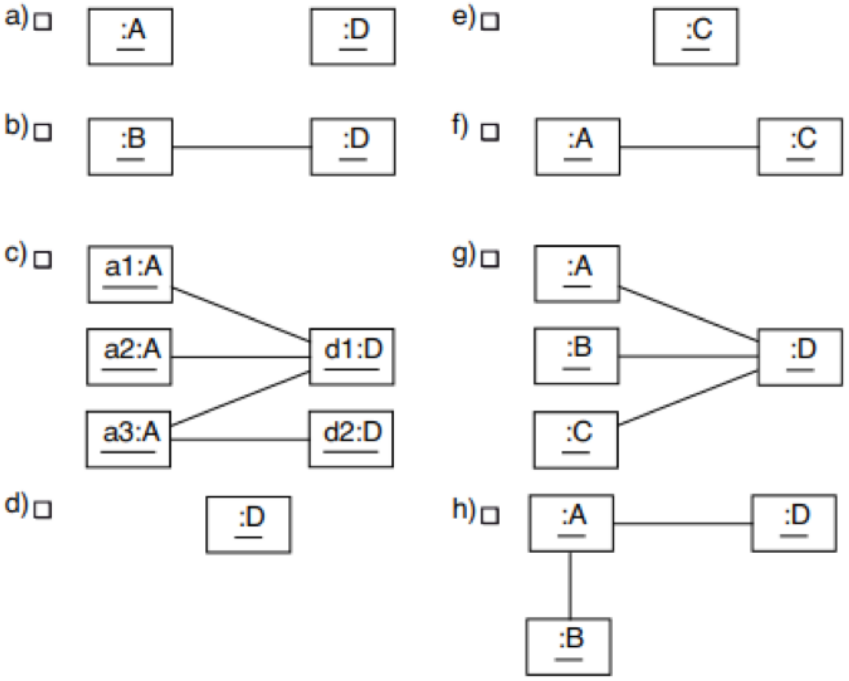
Deadline ist am Tag vor dem Praktikum 23:59 Uhr. Nicht, zu spät abgegebene Dateien oder nachträglich geänderte, werden mit 0% gewertet.

Aufgaben

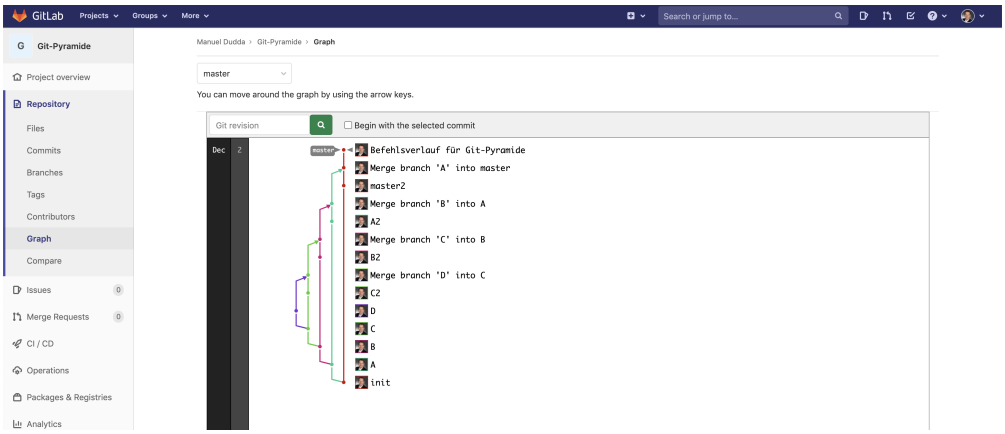
1. **Zusammenhang zwischen Klassen- und Objektdiagrammen I**
   Vorgegeben ist folgendes Klassendiagramm:



Geben Sie zu den folgenden Objektdiagrammen an, ob sie gültige Ausprägungen des Klassendiagramms sind oder nicht. Begründen Sie auch die Aussagen.



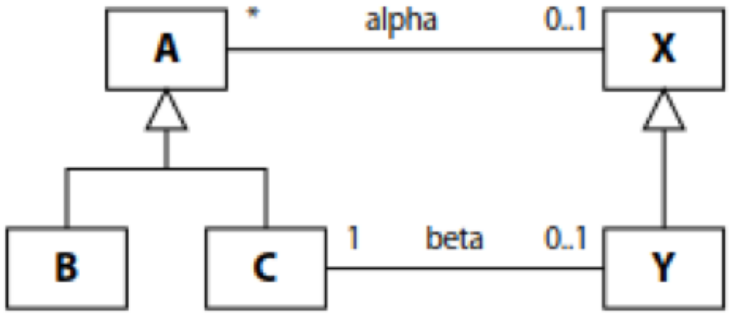
2. **Git-Pyramide:**
   Erstellen Sie ein neues Repository auf Gitlab der Hochschule: `https://gitlab.cs.hs-rm.de`. Benutzen Sie das Branching-Konzept um folgendes oder ähnliches Pyradmiden-förmiges Schaubild im Commit-Graphen herzustellen. Fügen Sie ihren Praktikumsleiter als Mitglied mit Lese-Berechtigung für das Repository hinzu. Notieren sie den Git-Befehlsverlauf für das „Aufbauen“ der Git-Pyrmaide und fügen Sie es als Text-Datei im Repository hinzu.



Hinweis: Die Befehle könnten wie folgt lauten:

```
git init # Initialisiere Rpeository
git remote add origin REPO_URL # entferntes Repository hinzufuegen
echo "init" >> README.md # aendere Dateien
git add README.md  # Fuege Datei zur Staging-Area hinzu
git commit -m "init" # Fuege Datei zum lokalen Rpeository hinzu
git branch A # Branch erstellen
git checkout A # wechsle zu Branch
echo "A" >> a.txt
git add a.txt
git commit -m "A"
git branch B
git checkout B
...
git merge B
git checkout master
echo "master2" >> README.md
git add README.md
git commit -m "master2"
git merge A
...
git commit -m "Befehlsverlauf fuer Git-Pyramide"
git push origin master
```

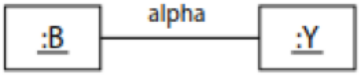
3. **Zusammenhang zwischen Klassen- und Objektdiagrammen II**
Vorgegeben ist folgendes Klassendiagramm:



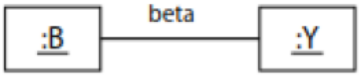
Geben Sie zu den folgenden Objektdiagrammen an, ob sie gültige Ausprägungen des Klassendiagramms sind oder nicht. Begründen Sie auch die Aussagen.

a) ☐

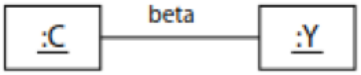
b) ☐



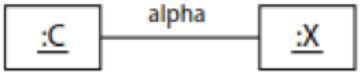
c) ☐



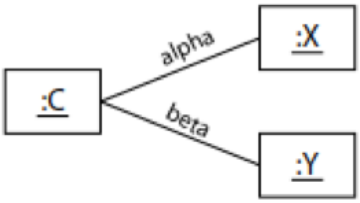
d) ☐



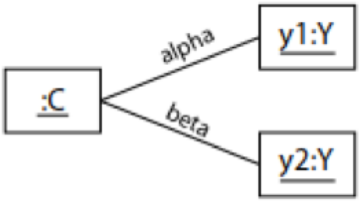
e) ☐



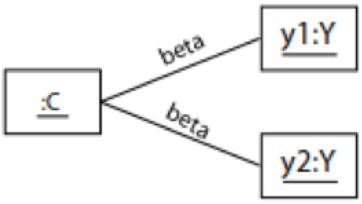
f) ☐



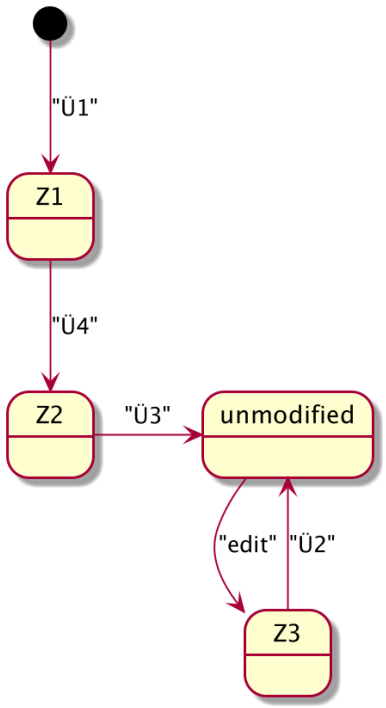
g) ☐



h) ☐



4. **Git Zustandsdiagramm: (freiwillig)**
   Betrachte das Zustandsdiagramm zum GIT-Workflow und vervollständige es um die Zustände (Z1 – Z3) und Übergänge (Ü1 – Ü4). Beachte: Nicht alle Zustände und Übergänge werden benötigt.



   (a) Mögliche Zustände (Z1 – Z3): „staged“ , „workspaced“ , „created“ , „Endzustand“ , Übergänge (Ü1 – Ü4): „git add“ , „git diff“ , „git init“ , „git commit“ , „git rm“ , „git checkout“

   (b) Mit welchem (Kurz-)Befehl(en) kann man Dateien aus dem „untracked“ -Zustand in den „unmodified“ -Zustand heben? Wenn nötig, ergänze den Übergang im Zustandsdiagramm

   (c) Füge den Zustand „tracked“ ein. Nutze dafür Unterzustände, sodass „tracked“ zwei Unterzustände beinhaltet.

   (d) Füge den Zustand „ignored“ ein. Mit welchem Befehl(en)/Vorgang wird dieser Zustand erreicht?