Übungsserie 10

Aufgabe 1: Adjazenz-Liste und Adjazenz-Matrix

Würden Sie in den folgenden Fällen eine *Adjazenz-Liste* oder eine *Adjazenz-Matrix* verwenden? Begründen Sie ihre Antwort aufgrund Folie 13 'Performance'.

a) Der Graph hat 10’000 Knoten und 20’000 Kanten und es ist wichtig so wenig Platz wie möglich zu verwenden.
b) Der Graph hat 10’000 Knoten und 20'000'000 Kanten und es ist wichtig, dass das Löschen von Knoten so schnell wie möglich ist.
c) Es ist wichtig die Anfrage auf *Nachbarschaft* so schnell wie möglich zu beantworten. Der verwendete Platz spielt keine Rolle.

Aufgabe 2: Implementation Graph

Es sollen in einer Graphen-Klasse die Methoden
`Graph.areAdjacent()`|`are_adjacent()` resp. `Graph.opposite()`implementiert werden.
Die Ausgangslage liegt auf ILIAS.
Es müssen nur in der Datei `Graph.java` resp. `graph.py` die beiden mit *"TODO"* markierten Methoden implementiert werden.

Hinweise:

- Es sollen jeweils nur die ADT-Graphen-Methoden gemäss der Skript-Folie 9 verwendet werden (Ausnahme: Bei Java kann anstelle von `Graph.incidentEdges()` die entsprechende Methode auf dem Vertex verwendet werden, z.B. `vi.getIncidentEdges()`)
- Die bestehenden Datei-Namen und weitere Definitionen dürfen nicht verändert werden (Package-Deklarationen, Imports, Klassennamen, Attribute, Methoden, etc.).
  Dies bedeutet somit insbesondere auch, dass sich die Dateien bei Java im Verzeichnis `uebung10/as/aufgabe02` befinden resp. bei Python im Verzeichnis `uebung10/al/aufgabe02`.

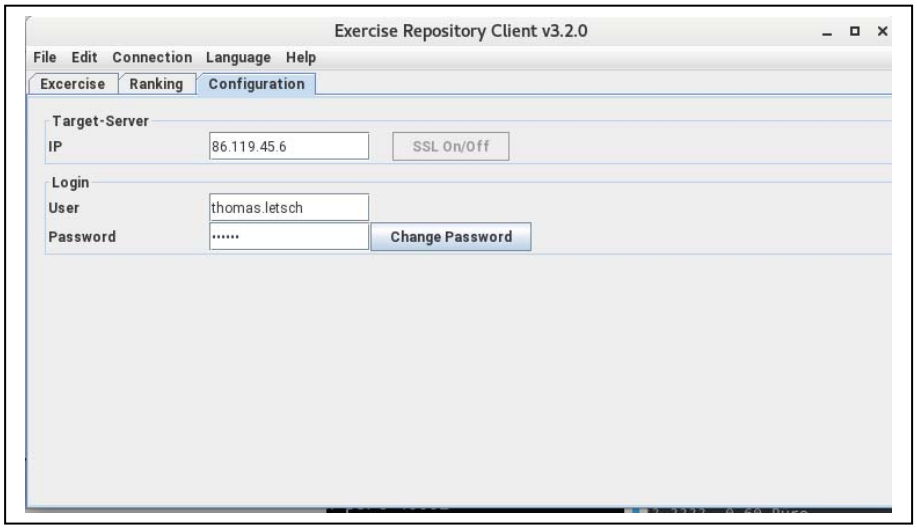
Testat

Diese Aufgabe 2 wird als Testat abgegeben.
Geprüft wird die Funktionalität der Methoden:

- `opposite(Vertex, Edge)`
- `areAdjacent(Vertex, Vertex)`

Abgabe

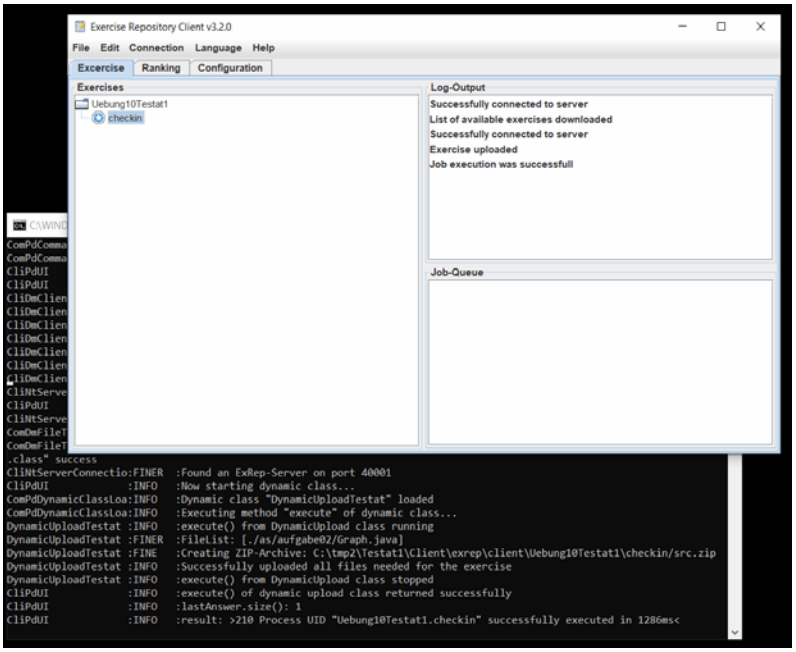
- Im Verzeichnis *uebung10* das Batch- resp. Shell-Script *checkin_** zur Ausführung bringen (entsprechend der Umgebung resp. Betriebssystem).
- Im Tab ***Configuration*** die Benutzer-Daten eintragen.
  User und *Password* sind jeweils *vorname.nachname* gemäss HSLU-EMail-Adresse (alles vor dem @, z.B.: *thomas.letsch* oder auch *hans.muster.01*).



Hinweise:
  - Es ist keine Bestätigung in Form von Return/Enter oder eines OK-Buttons nötig.
  - Falls Text-Feld *disabled* : Focus auf anderes Window und dann wieder zurück.

- Wechsel zum Tab ***Exercises.***
- Die Verbindung zum Server aufbauen mit Menü:***Connection>Connect***
  Es erscheint ein Ordner *Uebung10Testat1* mit einem Eintrag *checkin .*
- Zum Passwort-Wechsel zurück in den Tab ***Configuration***: Button ***Change Password***
  Hinweis: Ein *Change Password* kann erst nach einem erfolgreichem, initialen Login durchgeführt werden.

- Zum Upload zurück in den Tab ***Exercises*** und dort den Eintrag ***checkin*** unter *Uebung10Testat1* selektieren und:
  → Kontext-Menü (rechte Maus): ***Upload***



Auf dem Server wird dann ein Test durchgeführt mit *GraphTest.java* resp. *graph_test.py.*

Wichtig:
In der Pane *Log-Output* muss stehen: *Job execution was successfull*

Sonst: Den *exit code* im Log im Terminal-Window beachten.

- exit code: *1* → Kompilations-Fehler
- exit code: *11* oder *22* → Gemäss *GraphTest.java* resp. *graph_test.py*

Hinweise:

- Man kann zum Test bereits mit der Ausgangslage schon einen Check-In machen. Gemäss *GraphTest.java* resp. *graph_test.py* erfolgt dann zwar ein Fehler (*exit code 11*), aber der Check-In kann so schon mal getestet werden.
- Man kann beliegig of *einchecken*. Der letzte Check-In gilt dann als Testat-Abgabe.
- Die Server-Infrastruktur läuft jeweils von 08:00-23:00 Uhr.

Abgabe-Termine:

- Gruppe Vorlesung Dienstag: Mo 27.11.23 23:00 Uhr
- Gruppe Coaching Freitag: Do 30.11.23 23:00 Uhr