OOAD – 11

a) Testdaten einspielen

- Art: neue Art TESTDATA hinzugefügt
- Klasse TestAction erstellt:
    - -testdata
    - +Testdata(TaskList)
- ActionFactory: create erweitert um TESTDATA case
- Reducer: reduceIntern erweitert um if(instanceof TestdataAction)
- State: setTaskList

b) Task als finished markieren

- Art hinzugefügt
- Klasse FinishAction erstellt
- TextIO: finish erweitert
- ActionFactory: create erweitert um COMPLETE case
- Reducer: reduceIntern erweitert um if(instanceof FinishAction)

c) Tasks für einen Verantwortlichen anzeigen lassen

- Art: SHOW_RESPONSIBILITIES hinzugefügt
- ActionFactory: create erweitert um SHOWRESPONSIBILITIES case
- Klasse ShowResponsibilitiesAction erstellt:
    - – String responsible
    - + ShowResponsibilitiesAction(string)
- TextIO: showresponsibilities() erweitert
- Reducer: reduceIntern erweitert um if(instanceof ShowResponsibilitiesAction)

d) Tasks schützen

- Art: neue Art PROTECT hinzugefügt
- ActionFactory: create erweitert um PROTECT case und protectTasks()
- Klasse ProtectAction erstellt:
    - –int [] protectIds
    - +ProtectAction(int [] ids)
- State: Set<Integer> protectedSet hinzugefügt
- State: clone verändert
- State: protect() hinzugefügt
- Reducer: reduceIntern erweitert um if (instanceof ProtectAction)
- State: delete erweitert

e) Benutzerprüfung

- Klasse AllowedStore erstellt:
    - Erbt von AbstractDecoratorStore
    - Überschreibt dispatch(Action)

- TextIO: store zu AllowedStore geändert
- Alternativer Lösungsvorschlag: Decorator Patter auf Reducer anwenden

f) Alle Aufgaben eines Bearbeiters einem anderen zuweisen

- Art: neue Arte REARRANGE hinzugefügt
- Klasse RearrangeAction erstellt:
    - +RearrangeAction(String…)
- ActionFactory: create erweitert um REARRANGE case und createRearrangeAction() Methode
- Reducer: reduce erweitert um if(instanceof RearrngeAction)
- AllowedStore: dispatch erweitert