Aufgabe 12: GUI / MVC sowie (Vererbung / Collections / IO … aus Aufgabe 02)

Diese Aufgabe basiert auf der zweiten Angabe (*Mitarbeiter* / *Firma*). Es soll eine grafische Anwendung unter Einhaltung des **M**odel-**V**iew-**C**ontroller-Prinzips erstellt werden.

Dazu bitte ein neues Projekt mit zwei Packages erstellen:

- *data*: enthält alle Klassen des Models
  (Mitarbeiter … Firma / die eigenen Exceptions / Comparator-Klassen, …)
- *gui*: enthält alle Klassen der grafischen Oberfläche (Fenster / Listener / …)

Grafische Oberfläche – Hauptfenster

Das **Hauptfenster** enthält eine *JCheckBox* „nur Abteilungsleiter anzeigen“, ein Feld/Label welches die Gesamtanzahl aller Mitarbeiter der Firma angibt und eine Liste der der Mitarbeiter in Textform (z.B.: in Textform als *JTextArea* – ev. eingebettet in eine *JScrollPane* oder etwas anspruchsvoller in Form einer *JList* – lernen wir später).

Ist die **Checkbox** ausgewählt, dann werden nur Abteilungsleiter angezeigt, sonst alle Mitarbeiter.

Die Textanzeige dient vorerst zur Funktionskontrolle.

Weiters gibt es im Hauptfenster zwei **Menüs** mit folgenden Möglichkeiten:

Datei
- Laden
- Speichern

Mitarbeiter
- Anstellen
- Entlassen
- Suchen

Dialoge zum Anstellen / Entlassen / Suchen eines Mitarbeiters:

Bei Auswahl dieser Menüpunkte wird ein Mitarbeiter in die Liste (Model) übernommen oder daraus entfernt bzw. gesucht. Dazu wird jeweils ein eigenes (Eingabe)Fenster geöffnet…

Anstellen:

Dieses Fenster enthält die für alle Mitarbeiter benötigten Eingabefelder, sowie eine Auswahlmöglichkeit um welche Art von Mitarbeiter (Angestellter, Abteilungsleiter,…) es sich handelt (*JRadioButton* oder *JComboBox*). Je nach gewählter Mitarbeiterart sind nur die relevanten Felder eingabebereit.

Hat der Benutzer alle Felder befüllt und die Eingabe mittels „Ok“-Button bestätigt, wird versucht ein entsprechendes Objekt zu erzeugen und ins Model einzufügen.

Im Fehlerfall erhält der Benutzer eine Rückmeldung (siehe Fehlermeldungen).

Entlassen:

Es wird eine Sozialversicherungsnummer eingegeben und das betreffende Objekt aus der Liste (dem Model) entfernt. Der Benutzer erhält auch hier eine Rückmeldung, ob die Aktion erfolgreich war bzw. eine Fehlermeldung!

Suchen:

Der Benutzer gibt einen Namen ein und das Suchergebnis wird angezeigt – ähnlich dem Einstellen-Fenster. (siehe auch *sucheMitarbeiter(name : String) : Mitarbeiter*)

Dialoge zum Laden / Speichern der Angestelltenliste:

Bei Auswahl einer dieser Menüpunkte öffnet sich ein entsprechender *JFileChooser*-Dialog wo ein Dateiname bzw. Pfad zum Speichern oder Laden eingegeben bzw. ausgewählt werden kann.

Die Laden- und Speichern-Funktion ist im Model (also hier in der *Firma*-Klasse) mittels Serialisierung zu realisieren. Eventuell auftretende Fehler (*IOExceptions*) werden an die Oberfläche (View) weitergeleitet! (siehe Fehlermeldungen).

Es ist das MVC-Prinzip einzuhalten!

D.h. es gibt eine klare Trennung von **Model** und GUI (und hier wiederum zischen **View** und **Controller**). Alle grundlegenden Funktionen, wie das Erstellen, Hinzufügen oder Entfernen von Objekten, das Sortieren, das Generieren von Ausgabelisten in Textform, usw. ist in den geeigneten Modelklassen zu implementieren.

Mögliche Fehler werden nicht durch direkte Überprüfungen im Eingabefenster erkannt (ausgenommen ganz grundlegende Prüfungen, z.B.: Wurde eine Zahl eingegeben?), sondern die Daten werden an die entsprechenden Model-Klassen übergeben (Konstruktoren / einstellen-Methode / entlassen-Methode).

Wenn ein Konstruktor oder eine Methode einen Fehler erkennt, wird ohnehin ein geeigneter Rückgabewert geliefert oder eine Exception geworfen, worauf in der GUI-Klasse reagiert werden muss und die Ausgabe einer Fehlermeldung erfolgt.

Fehlermeldungen:

Sämtliche Fehler (*Exceptions*) – welche vermutlich im Controller „gefangen“ werden - sind an die View-Klasse weiterzureichen und dort auszugeben - z.B. mittels *JOptionPane* :
`JOptionPane.showMessageDialog(…);`

Erweiterung:

Überlege Dir, wie eine nach unterschiedlichen Kriterien sortierte Liste angefordert, generiert und dargestellt werden könnte.