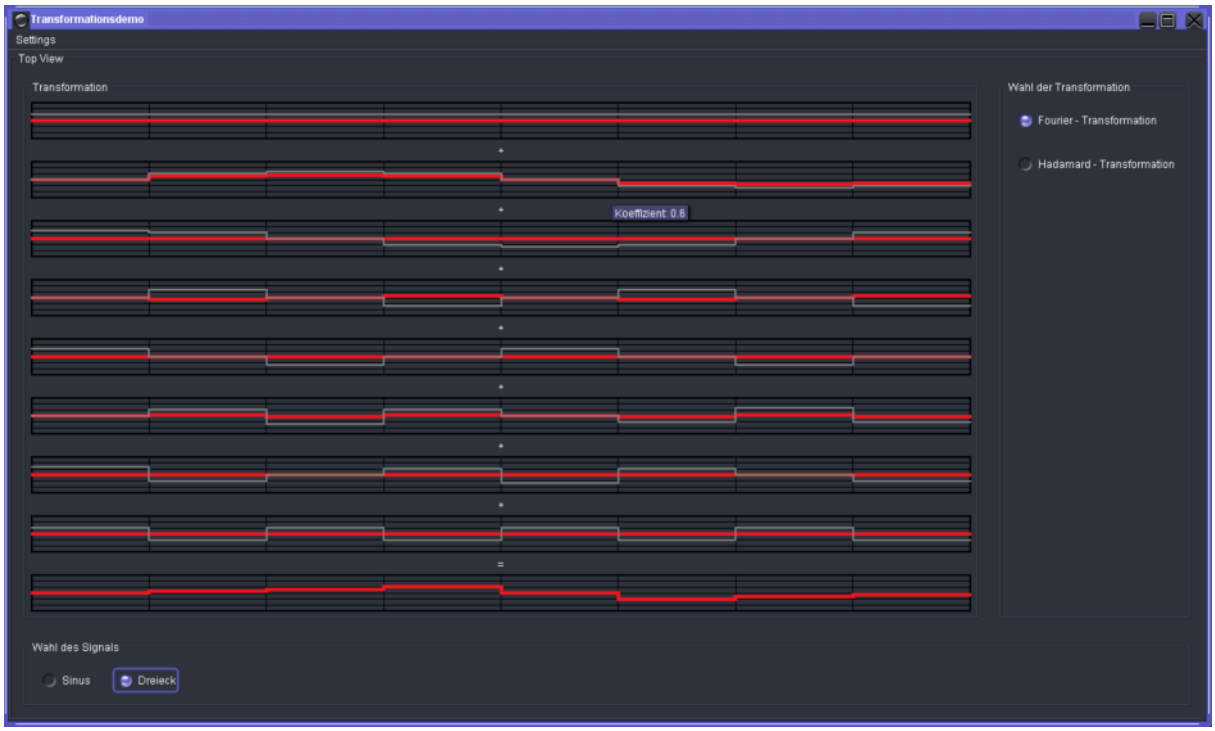
JAVA PRÜFUNG 2

Bedingungen:

- Erlaubte Hilfsmittel: Unterrichtsunterlagen, Java Buch und Übungen.
- **Die Prüfung ist schrittweise, gemäss Aufgabenstellung lokal auf Ihrem Computer zu lösen. Kopieren Sie zu diesem Zwecke den gesamten Ordner Pruefung2 auf Ihre lokale Harddisk und importieren Sie das Projekt in Eclipse. Am Ende der Prüfung ist der Ordner src abzugeben, Anweisungen folgen.**
- **Setzen Sie als erstes Ihren Namen und Vornamen in die Dateien.**
- Gegenseitiges Abschreiben in irgendeiner Form führt zur Note 1!
- Folgend Sie bei der Wahl von Variablen den Angaben in der Aufgabenstellung.
- **Die Beilage muss abgegeben werden!**

Beschreibung:

Transformationen nehmen in der Technik eine wichtige Position ein. Ziel dieser Prüfung ist es, eine Applikation zu programmieren, die erlaubt die Fourier- und die Hadamard – Transformation zu demonstrieren:



Mit der zu programmierenden Applikation lässt sich zeigen, wie eine Zeitfunktion durch einen Satz orthogonale Funktionen zusammengesetzt werden kann. Wohl bekannt ist die Fourier - Transformation respektive die Fourier- Reihe. Ebenso lässt sich die Zeitfunktion auf orthogonale Rechteckfunktionen abbilden. Diese Transformation ist als Hadamard – Transformation bekannt.

View: Die ***TopView*** des User–Interfaces der Applikation ist mittels dreier Panel realisiert: Das ***PanelTransformPlot*** sitzt oben links und beinhaltet mittels der Klasse ***PanelPlot*** realisierte Plots der 8 Funktionen sowie den Plot des gewählten Signals. Das ***PanelTransformWahl*** sitzt neben den Plots und erlaubt die Wahl der gewünschten Transformation. Das ***PanelSignalWahl*** sitzt unter dem Panel mit den Plots und erstreckt sich horizontal über zwei Zellen.

PanelTransformPlot wächst sowohl horizontal wie vertikal. ***PanelTransformWahl*** wächst nur vertikal und hat ein leeres ***JLabel*** um den vertikalen Zuwachs aufzunehmen. ***PanelSignalWahl*** wächst nur in horizontaler Richtung und hat ein leeres ***JLabel*** um den horizontalen Zuwachs aufzunehmen.

Menu - Bar: Die Applikation verfügt über ein Menu mit ***MenuItem*** *Farbe*, um die Farbe der Plots zu ändern und über ein ***MenuItem*** *Exit* um das Programm zu schliessen.

Model: Dem ***Model*** kommt die Berechnung der jeweiligen Transformation zu. Dazu die Methode ***transform***(), die jeweils beim Ändern der Signalform resp. beim Ändern der Transformation aufgerufen wird. Die Art der Transformation resp. die Art des Signals wird mittels entsprechender Setter–Methoden gesetzt.

Controller: Der Controller enthält die wenigen Methoden zur Steuerung des Programms.

Das Programm ist im bekannten Model-View-Controller Pattern zu programmieren. Die Details können der Beschreibung im Code sowie dem Klassendiagramm entnommen werden.

Die nachfolgende Aufgabenstellung führt sie schrittweise zum Ziel. Halten sie sich an die Vorgaben bezüglich Methoden- und Variabelennamen. **Achten Sie darauf, dass ihr Code jeweils ohne Fehler kompiliert werden kann. Fragen sie im Notfalle den Dozenten.**

Aufgabe 1: *Klassen PanelSignalWahl, PanelTransformWahl* (je 16 Pte.)

Als erstes wollen wir die Klassen ***PanelSignalWahl***, ***PanelTransformWahl*** schreiben. Die Klassen beheimaten die User-Interface Elemente zur Wahl von Signalform und Transformation.

a) Implementieren Sie die Klassen gemäss Beschreibung im Code.

Aufgabe 2: Klasse *PanelTransformPlot* (15 Pte.)

Als nächstes wollen wir die Klasse ***PanelTransformPlot*** schreiben. Die Klasse ***PanelTransformPlot*** beheimatet die Panel mit den Plots.

a) Implementieren Sie die Klasse gemäss Beschreibung im Code.

Aufgabe 3: Klasse *TopView* (9 Pte.)

Die Klasse ***TopView*** beheimatet die drei nun programmierten Panels mit den Plots und den Wahlmöglichkeiten.

a) Implementieren Sie die Klasse gemäss Beschreibung im Code.
b) Überprüfen Sie die Applikation auf richtiges Layout hin.

Aufgabe 4: Klasse *Controller* (5 Pte.)

Die Klasse Controller beinhaltete die Geschäftslogik des Programms und umfasst nur wenige Methoden.

a) Implementieren Sie die Klasse gemäss Beschreibung im Code.

Aufgabe 5: Klasse *Model* (37 Pte.)

Die Klasse ***Model*** hat die Aufgabe, die Transformationen zu berechnen. Dazu sind die beiden Transformationsmatrizen ***fourier*** und ***hadamard*** vordefiniert. Je nach gewählter Transformation wird nun das Attribut ***transformMatrix*** gleich der entsprechenden Matrix gesetzt. Die Transformation geschieht somit immer aufgrund von ***transformMatrix*** und berechnet wie folgt die Koeffizienten der Amplituden:

$$c_{oeff}[i] = \frac{1}{8}\sum_{j=0}^{7} s_{ignal}[j] \cdot t_{ransformMatrix}[i][j]$$

Diese Koeffizienten gilt es für i = 0 ... 7 zu berechnen. Das Sinus- und Dreiecksignal sind für i = 0...7 gegeben durch:

Sinus: $s_{ignal}[i] = \sqrt{2} \cdot \sin(2 \cdot \pi \cdot \frac{i}{8})$

Dreieck: $s_{ignal}[i] = \begin{cases} \frac{i}{3} & 0 \le i \le 3 \\ 0 & i = 4 \\ \frac{i-5}{3} - 1 & 5 \le i \le 7 \end{cases}$

a) Implementieren Sie die Klasse gemäss Beschreibung im Code und obiger Formeln.
b) Überprüfen Sie die Applikation auf richtiges Funktionieren hin.

Aufgabe 6: Challenge: Klasse *MenuBar* (16 Pte.)

Die Klasse ***MenuBar*** implementiert die eingangs erwähnte Menu–Struktur. Unter *Settings* kann die Farbe der Plots eingestellt oder die Applikation geschlossen werden. Zur Auswahl der Farbe steht die Klasse ***ColorPalette*** zur Verfügung, die von ***JDialog*** erbt. Nachdem der Dialog erzeugt wurde, kann er mittels setVisible(true) gezeigt werden. Nach der Farbwahl mittels einfachem Klick verschwindet der Dialog und die Farbe kann mittels ***getColor***() aus dem Objekt ausgelesen werden.

a) Implementieren Sie in die Klasse ***MenuBar*** gemäss Beschreibung im Code