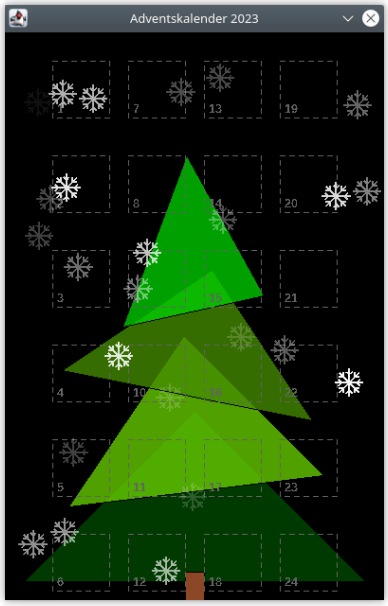
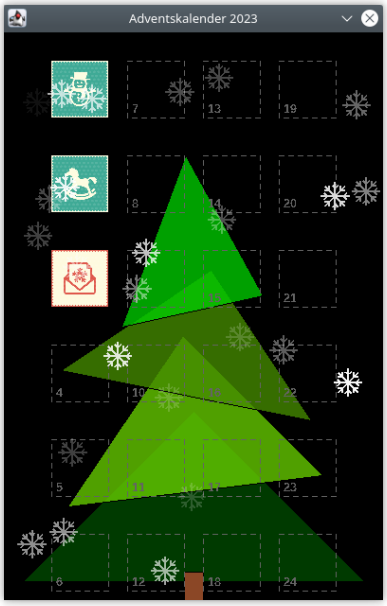
Projektblatt P11 (15 + 12* + 5*P)

Abgabe: Freitag 1. Dezember 2023, 12:00h
Entpacken Sie zunächst die Archiv-Datei `vorgaben-p11.zip`, in der sich die Rahmendateien für die zu lösenden Aufgabe sowie ein Unterverzeichnis mit Bildern befinden. Ergänzen Sie die Dateien durch Ihre Lösungen gemäß der Aufgabenstellung unten. Der hinzuzufügende Java-Sourcecode sollte **syntaktisch richtig** und **vollständig formatiert** sein. Alle Dateien sollten am Ende fehlerfrei übersetzt werden können.
Verpacken Sie die `.java` Dateien für Ihre Abgabe in einem ZIP-Archiv mit dem Namen `IhrNachname.IhrVorname.P11.zip`, welches Sie auf Ilias hochladen.

Aufgabe 1: 15 + 12* + 5* Punkte

In dieser Aufgabe sollen Sie einen elektronischen Adventskalender programmieren. Hierzu sollen Sie zunächst ein Hintergrundbild erstellen und dann "Türen" auf diesem Hintergrund platzieren, bei deren Öffnung (durch einen Mausklick) jeweils ein Bild gezeigt wird. Der folgende Screenshot zeigt Ihnen ein Beispiel eines Adventskalenders, einmal direkt nach dem Öffnen und daneben mit drei geöffneten Türchen.





Hierzu sind folgende Klassen (teilweise) vorgegeben:

- Die Klasse `Adventskalender` definiert einen Rahmen für das Programm in Form einer GUI, die in der `main`-Methode erzeugt und angezeigt wird. Die GUI besitzt nur eine wesentliche Komponente: eine Instanz der Klasse `Adventspanel`, die von der Klasse `JPanel` abgeleitet ist und in der alle Bestandteile des Adventskalender zusammengestellt und gezeichnet werden sollen.

- Die Klasse `ImageLoader` stellt eine statische Methode zur Verfügung, das alle Dateien in einem Verzeichnis einliest und in Form eines Felds von Instanzen der Klasse `BufferedImage` zurückgibt. Hierzu muss der Methode der (absolute oder relative) Pfad zu dem Verzeichnis übergeben werden. Die Dateien in dem Verzeichnis dürfen ausschließlich Bilder in einem von Java lesbaren Format sein (png, jpg, gif, ...). Die Archivdatei beinhaltet ein solches Verzeichnis mit 24 Bildern.

- Die Klasse `Adventspanel` gibt ebenfalls bereits einen Rahmen für den eigentlichen Adventskalender vor:

  - In dem statischen Code werden alle Bilder aus dem Unterverzeichnis "bilder" geladen. Diese Unterverzeichnis muss sich hierzu in dem Verzeichnis befinden, aus dem das Programm gestartet wurde.
  - Die "Türen" des Adventskalenders sollen durch Instanzen der inneren Klasse "Tuerchen" repräsentiert werden. Diese Klasse stellt ein Rechteck dar, welches zusätzlich mit einem Bild und einem Datum ausgestattet ist und eine einfache `paint`-Methode besitzt, die das Objekt als Rechteck zeichnet.
  - Im Konstruktor der Klasse wird die Größe der Komponente gesetzt und eine Methode aufgerufen,in der später die Türchen generiert werden sollen.
  - Die `paintComponent`-Methode der Klasse ist aktuell so überschrieben, dass ein schwarzer Hintergrund sowie das Hintergrundbild und darauf dann die Türchen gezeichnet werden.

- Die Klasse `Hintergrundbild` zeigt Ihnen beispielhaft, wie ein Hintergrundbild aufgebaut und gezeichnet werden kann. Die Klasse hat

hierzu zwei Felder `teile` und `farben` vom Typ `Shape` bzw. `Color`. Das Feld `teile` kann Objekte aufnehmen, die einer der Klassen angehören, die das Interface `Shape` implementieren, also z.B. `Rectangle`, `GeneralPath`, usw. Hierdurch lassen sich verschiedene, individuelle Formen in einem Bild zusammenfassen. Um jedem Objekt eine eigene Füllfarbe zuzuordnen, wird das Feld `farben` verwendet. Die `paint`-Methode in der Klasse zeichnet dann diese Formen in den zugeordneten Farben.

Anmerkung: Grundsätzlich sollte man zur Repräsentation der einzelnen Bestandteile besser eine eigene (innere) Klasse schaffen, die die Eigenschaften eines Objekts (hier: Form + Farbe) bündelt. Das wurde hier unterlassen, um den Zugriff etwas einfacher zu gestalten. Für eine geeignete Anpassung der Klasse erhalten Sie 5 Bonuspunkte.

In der Klasse sind zwei statische Variablen `dreieck` und `quadrat` definiert, die zwei Grundformen repräsentieren und die in dem statischen Code initialisiert werden. Im Konstruktor werden dann zu Beispielzwecken ein Dreieck und zwei Rechtecke aus diesen Grundformen "abgeleitet". Hierzu werden affine Transformationen verwendet. Bei Bedarf können Sie sich an dieser Vorgehensweise orientieren, um komplexere Objekte aus den Grundformen zusammenzusetzen. Der Tannenbaum im Bild oben ist auf diese Weise aus mehreren Dreiecken erstellt worden. Dazu wurden für die einzelnen Teile z.T. transparente Farben definiert.

Ergänzen Sie nun die Klassen `Hintergrundbild` und `Adventspanel` wie im Folgenden beschrieben:

(a) 15P

Erweitern Sie nun zunächst die Klasse `Hintergrundbild`, indem Sie dort (im Konstruktor oder in zusätzlichen Methoden) grafische Objekte anlegen, die Sie in dem Feld `teile` speichern. Für jedes (Teil-)Objekt muss eine Farbe in dem Feld `farben` angelegt werden. Alternativ dürfen Sie die Klasse so umschreiben, dass Sie eine zusätzliche, innere Klasse anlegen und verwenden (d.h. dann auch ein Feld mit diesen Objekten anzulegen und die `paint`-Methode entsprechend umzuschreiben).

Den im Konstruktor definierten Code dürfen Sie hierbei löschen oder verändern. Sie dürfen ggf. auch weitere Grundformen anlegen. Alternativ können Sie auch weitere Objekte direkt definieren, also mit absoluten Koordinaten und ohne affine Transformationen.

Ihr Bild sollte aber mindestens fünf weitere Objekte (anstelle der drei vorgegebenen Objekte bzw. zusätzlich zu diesen) enthalten. Verwenden Sie zu deren Erzeugung verschiedene Arten von Shape-Objekten und unterschiedliche affine Transformationen.

Gestalten Sie das Aussehen des Hintergrundbilds weihnachtlich.

(b) 8*P

Erweitern Sie nun die innere Klasse `Tuerchen` (in der Klasse `Adventspanel`) wie folgt:

- Fügen Sie ein Zustandsattribut hinzu, welches angibt, ob die Tür geöffnet wurde oder nicht. Initial soll eine Tür immer geschlossen sein.
- Vervollständigen Sie die Methode `oeffnen`, welche überprüfen soll, ob die Tür geöffnet werden darf. Überprüfen Sie hierzu, ob das aktuelle Datum zeitlich nach dem Öffnungsdatum der Tür liegt. Ist dies der Fall, soll der Zustand des `Tuerchen` Objekts entsprechend geändert werden.
- Ändern Sie die Implementierung der `paintComponent`-Methode so ab, dass das `Tuerchen` Objekt wie folgt gezeichnet wird:
  Ist die Tür in geschlossenem Zustand, soll der Rahmen der Tür

(also das begrenzende Rechteck) durch eine gestrichelte Linie gezeichnet werden, die auf dem Hintergrund erkennbar sein sollte. Darüberhinaus soll der zugehörige Tag ($\in \{1, \ldots, 24\}$) auf die Tür geschrieben werden. Hinweis: Die Zahl für den Tag erhalten Sie, in dem Sie die Methode `getDayOfMonth` auf das Attribut `oeffnenAm` anwenden. Ist die Tür in offenem Zustand, soll das zugehörige Bild aus der Instanz in diesen Rahmen (also an die Stelle der Tür mit der Größe der Tür) gezeichnet werden.

(c) 4*P

Implementieren Sie nun einen `MouseListener` in Form einer weiteren inneren Klasse in der Klasse `Adventspanel`. Bei einem Mausklick auf das Panel soll dann überprüft werden, ob der Mausklick auf einem der Türen erfolgte. Ist dies der Fall, soll die betreffende Tür geöffnet (also der Zustand geändert) und das Panel neu gezeichnet werden. Hinweis: Nutzen Sie das Attribut `tuerchen` der Klasse `Adventspanel` und die von der Klasse `Rectangle` geerbte Methode `contains`, um zu überprüfen, ob die Koordinaten des Mausklicks inerhalb eines der `Tuerchen` Objekte liegen.

Ergänzen Sie den Konstruktor der Klasse `Adventspanel` so, dass eine Instanz des Listeners generiert und bei diesem Panel registriert wird.