Übung 1

Aufgabe 1:

Schreiben Sie ein Java Programm, dass ein Swing Fenster öffnet, in das eine Linie von links oben nach rechts unten und eine Linie von rechts oben nach links unten gezeichnet wird. Dazu soll die Fenstergröße dynamisch ermittelt werden, in der JComponent können Sie hierzu mittels getSize() die Größe abfragen.

Aufgabe 2:

Schreiben Sie ein Java Programm, dass ein Swing Fenster öffnet und ausgefüllte Rechtecke in das Fenster zeichnet. Es sollen m-viele Rechtecke in einer Zeile und n-viele Zeilen untereinander gezeichnet werden. Zwischen den Rechtecken soll immer 2 Pixel Platz gelassen werden. Die beiden Werte für m und n werden im Konstruktor Ihrer Klasse angegeben. Zum Errechnen der Größe der Rechtecke benötigen Sie die Größe der JComponent. Diese berechnen Sie mittels der getSize() Methode.

Übung 2

Aufgabe 1:

Stellen Sie Ihre beiden Programme aus der ersten Übung auf das MVC Muster um. Die Rechtecke der zweiten Aufgabe sollen zusätzlich farbig gezeichnet werden. Dazu sollen die Farben zufällig bestimmt werden, so dass jedes Rechteck eine andere Farbe besitzt.

Aufgabe 2:

Schreiben Sie ein Java Programm, dass ein Swing Fenster öffnet und die Deutschlandflagge (oder irgendeine andere Flagge) in das Swing Fenster zeichnet. Die Flagge soll an die Fenstergröße angepasst sein, die Größe ermitteln Sie mit getSize() Methode der JComponent Klasse.

Übung 3

Aufgabe 1:

Erweitern Sie Ihre Programme aus der zweiten Übung derart, dass Sie die Icons der Fenster und die Mousecursor verändern.

Aufgabe 2:

Das Fenster zur Darstellung der Deutschlandflagge (oder irgendeiner anderen Flagge) soll in der Mitte des Fensters den Namen des Landes, zu der die Flagge gehört, geschrieben werden.

Aufgabe 3:

Versuchen Sie bei Aufgabe 2 die Größe der Schrift an die Fenstergröße anzupassen, so dass bei größerem Fenster auch die Schrift größer wird. Wie üblich ermitteln Sie die Größe der JComponent mittels der getSize() Methode.

Übung 4

Aufgabe 1:

Erweitern Sie Ihr Programm zur Darstellung der Rechtecke derart, dass Sie einzelne Rechtecke mit der Maus anklicken können und auf dem Bildschirm (Konsole mittels System.out.println) die Position des Rechtecks ausgegeben wird, sprich Zeile und Spalte.

Aufgabe 2:

Implementieren Sie den „Schließen“ Button für Ihre Fenster.

Übung 5

Aufgabe 1:

Verändern Sie Ihr Programm zur Darstellung der Rechtecke derart, dass Sie die Maus über einem beliebigen Rechteck positionieren und durch Drehen des Mausrades die Farbe dieses Rechtecks ändern.

Aufgabe 2:

Schreiben Sie ein Java Programm (MVC Muster beachten), das ein Swing Fenster öffnet, in dem an einer zufälligen Position ein ausgefüllter Kreis gezeichnet wird. Sobald die Maus in den Kreis fährt, soll der Kreis an einer anderen Position gezeichnet werden.

Übung 6

Aufgabe 1:

Erweitern Sie Ihre Programme aus der letzten Übung derart, dass Sie Größe und Farben der Rechtecke bzw. des Kreises mittels Menus einstellen können.

Aufgabe 2:

Implementieren Sie den Heap- und Mergesort für int-Arrays. Testen Sie Ihre Programme mit Arrays der Größen 10, 100, 1000 und 10000. Dabei sollen die Arrays bereits sortiert, invers sortiert und mit zufälligen Zahlen belegt sein.

Übung 7

Aufgabe 1:

Erweitern Sie Ihr Programm mit dem Kreis, der immer an einer neuen Position gezeichnet wird, sobald der Mauszeiger ihn berührt derart, dass Sie über ein neues Menu ein modales Dialogfenster öffnen, in dem Sie neben einem Kreis noch weitere Formen einstellen können.

Aufgabe 2:

Wenn das Fenster geschlossen wird, soll ein Dialog erscheinen, der abfragt, ob das Fenster wirklich geschlossen werden soll.

Aufgabe 3:

Implementieren Sie eine Vektorklasse für double Zahlen. Fügen Sie der Implementierung eine push_front Methode hinzu, die am Anfang des Vektors ein neues Element hinzufügt.

Aufgabe 4:

Modifizieren Sie Ihre push_front Methode derart, dass sie analog wie die push_back Methode auch am Anfang mehr Elemente alloziert als notwendig sind.

Übung 8

Aufgabe 1:

Dort wo es möglich ist, ersetzen Sie die selbstgeschriebenen Dialogklassen durch JOptionPane Methodenaufrufe.

Aufgabe 2:

Implementieren Sie eine einfach und eine doppelt verkettete Liste als Generics. Die Listen sollen die Methoden push_font, push_back, pop_front, pop_back und isEmpty besitzen. Weiterhin soll die toString Methode überlagert werden, so dass bei System.out.println(l); die Liste l auf der Konsole ausgegeben wird.

Übung 9

Aufgabe 1:

Schreiben Sie ein Java Programm (MVC Muster beachten), das ein Swing Fenster öffnet, das zwei JSlider besitzt. Über diese beiden Slider soll ein Kreis auf dem Bildschirm verschoben werden können. Ein Slider stellt die horizontale, einer die vertikale Position ein.

Aufgabe 2:

Implementieren Sie jeweils eine search Methode für Ihre Listen und den Vektor.

Übung 10

Aufgabe 1:

Implementieren Sie einen binären Suchbaum, der Strings auf Integer abbildet. Der Suchbaum soll die Methoden insert und delete besitzen. Die toString Methode soll überlagert werden, so dass er Suchbaum auf der Konsole ausgegeben wird. Implementieren Sie sowohl eine iterative als auch eine rekursive Suchmethode.

Aufgabe 2:

Implementieren Sie für Ihren binären Suchbaum eine rekursive Methode nrOfNodes(), die die Anzahl der Knoten ermittelt und zurückgibt.

Aufgabe 3:

Implementieren Sie für Ihren binären Suchbaum eine rekursive Methode sumOfNodes(), die die Summe aller int Werte, die in den Knoten gespeichert sind, ermittelt und zurückgibt.

Übung 11

Aufgabe 1:

Schreiben Sie ein Java Programm (MVC Muster beachten), dass ein Swing Fenster öffnet, das an einer zufälligen Position einen Kreis zeichnet und nach einer kurzen Zeit den Kreis an einer neuen Position zeichnet. Wenn der Mauszeiger sich im Kreis befindet und die linke Maustaste gedrückt wird, soll ein Dialogfenster geöffnet werden, in dem „Treffer“ steht. Die Anzahl der bisherigen Treffer soll ebenfalls angezeigt werden.

Aufgabe 2:

Implementieren Sie eine Hashtabelle, die Strings auf float Werte abbildet.

Aufgabe 3:

Testen Sie Ihre Hashtabelle, indem Sie viele Werte eintragen. Protokollieren Sie die Häufigkeit von Kollisionen. Wie viele Kollisionen treten im Durchschnitt auf, wenn ein neues Element eingetragen wird und die Tabelle zu 40% (respektive 50%, 60%, 70%, 80%, 90% 95%) gefüllt ist?

Wichtig: Überlegen Sie sich zunächst, wie ein solcher Testaufbau stattfinden kann. Würden Sie bei einer Füllung von 40% einen Eintrag vornehmen und die Kollisionen zählen, wäre für den nächsten Eintrag die Füllung nicht mehr 40% sondern 40% + 1 Eintrag. **Dies soll nicht passieren!**
Führen Sie die Messungen mit unterschiedlichen Hashfunktionen durch.

Übung 12

Aufgabe 1:

Schreiben Sie ein Java Programm (MVC Muster beachten), das ein Swing Fenster öffnet, das eine SplitPane beinhaltet. Auf der einen Seite wird ein JTree dargestellt auf der anderen Seite einen Dialog. Der JTree soll einen Binärbaum darstellen, der Strings auf Integer Werte abbildet. In der Dialogseite befinden sich zwei Textfelder. In dem einen Textfeld kann man einen String angeben, in dem anderen Textfeld einen Integer Wert. Mittels eines „add“ Buttons wird das String/Integer Paar in den Binärbaum eingefügt und der JTree muss aktualisiert werden.