Prog 2

Objektorientierte Programmierung in Java

ÜBUNGSAUFGABEN

Übungsaufgabe

Verwaltung eines Marathonlaufs

A

Schreiben Sie eine Klasse `Laeufer`, die für einen Stadtmarathon Läufer darstellt. Ein `Laeufer` besitzt:

- einen Namen
- eine Startnummer
- ein Gewicht
- ein Alter
- soll ausgedruckt werden können
- soll erzeugt werden können

B

Schreiben Sie eine Klasse `LaufVerwaltung`, die es erlaubt, eine **beliebige** Menge von Läufern zu verwalten. Das bedeutet, durch diese Klasse sollen

- `Läufer` eingelesen werden können
- dabei bekommt jeder Läufer eine eindeutige **Startnummer** zugeordnet
- intern in einem (Vektor !!1!) gemerkt werden
- auf Bedarf die bisher enthaltenen `Läufer` ausgedruckt werden können
- Läufer mit einer bestimmten Startnummer wieder gelöscht werden können

Übungsaufgabe

Programmieren Sie eine Klasse Eieruhr. Eine Eieruhr kann mit einem bestimmten Integer-Wert erzeugt werden, der die "Laufzeit" der Eieruhr darstellt.

Nachdem eine Eieruhr dann ge - start() - et wurde, zählt sie diese Laufzeit sekundengenau bis 0 herunter und erwähnt dabei jede verstrichene Sekunden (genau **einmal** natürlich nur...).
Ist die Eieruhr dabei bei 0 angelangt, sagt sie "klingel".

Folgendes Programm:

```
public static void main(String[] args)
{
  Eieruhr eu = new Eieruhr(10);
  eu.start();
}
```

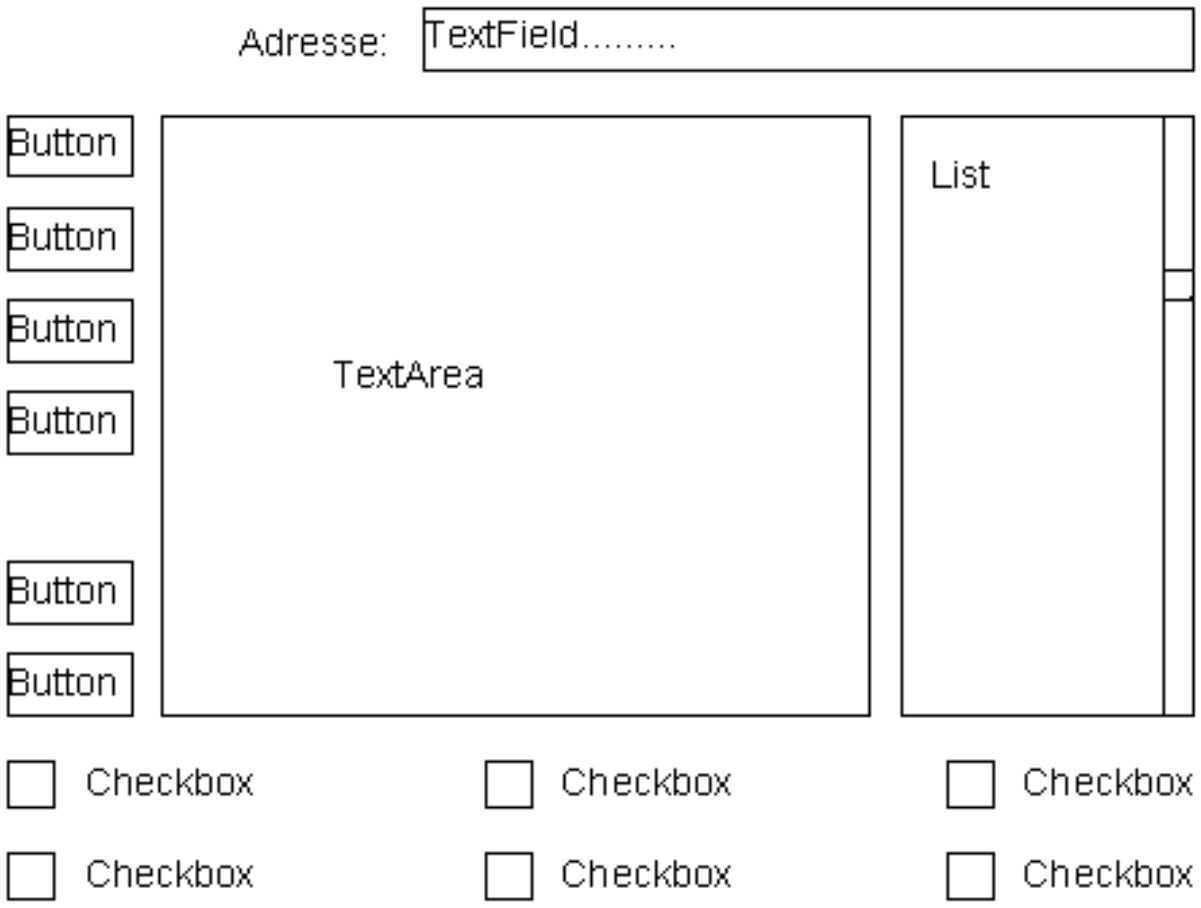
Soll das Ergebnis liefern:

```
tick - 9
tick - 8
tick - 7
tick - 6
tick - 5
tick - 4
tick - 3
tick - 2
tick - 1
tick - 0
klingel!
```

Übungsaufgabe

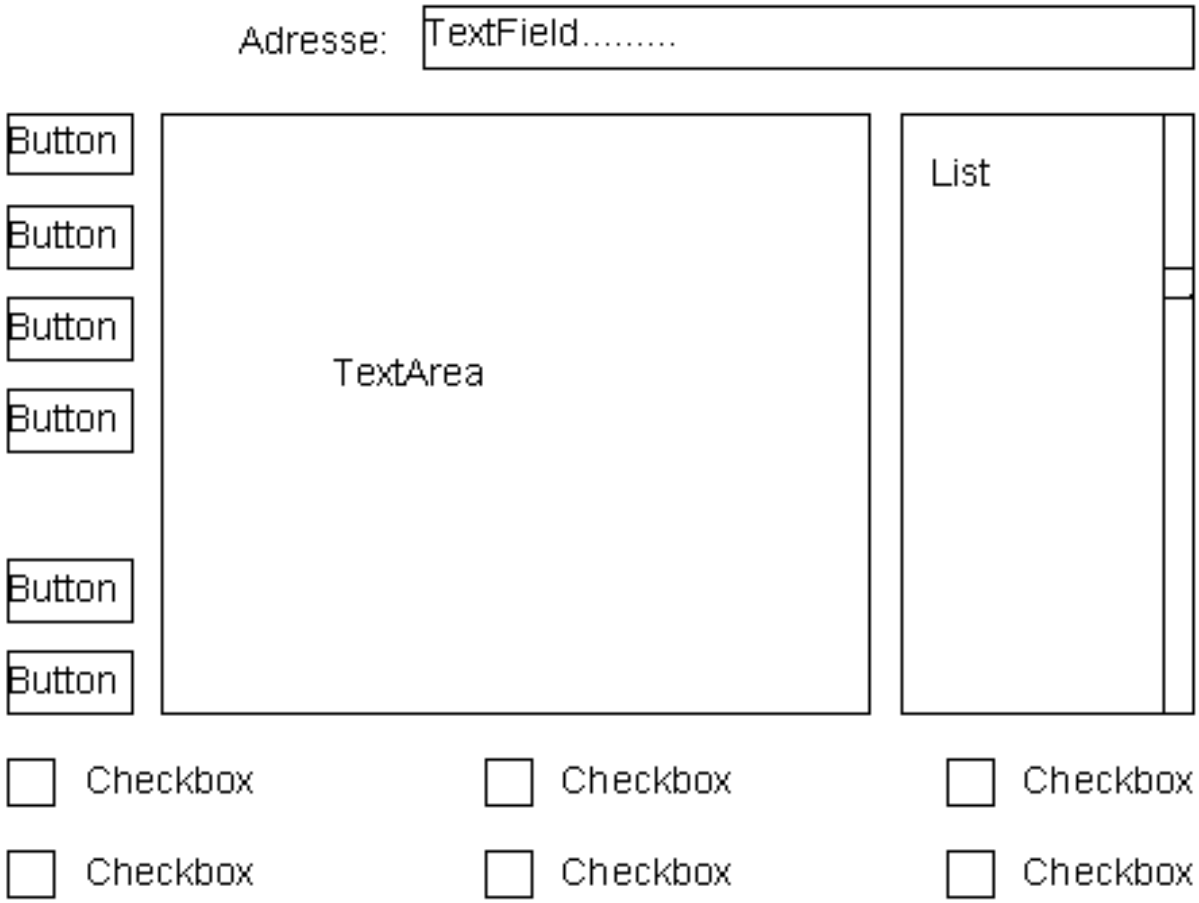
A Kombinieren Sie die Ihnen bekannten Layout-Manager, so dass das unten skizzierte Layout realisiert wird. Arbeiten Sie dabei nur mit Papier und Bleistift und nutzen Sie **keine** Software!

B Implementieren Sie das Layout in Java!



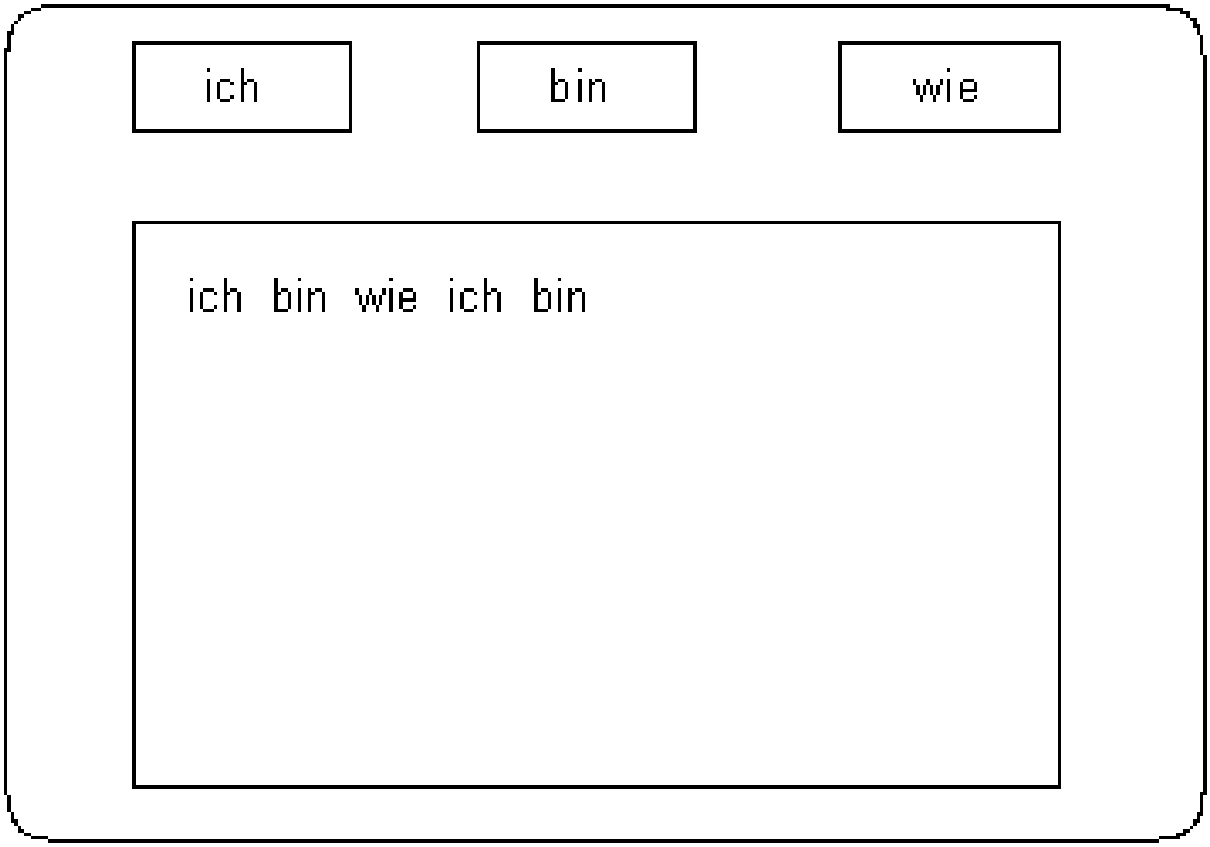
Übungsaufgabe

Implementieren Sie das Layout aus der letzten Übungsaufgabe nochmals mit Swing!



Übungsaufgabe

Implementieren Sie das unten dargestellte Programm in Swing!



Aufgaben: (zum Kapitel „Grafik“ und „paul ist traurig“)

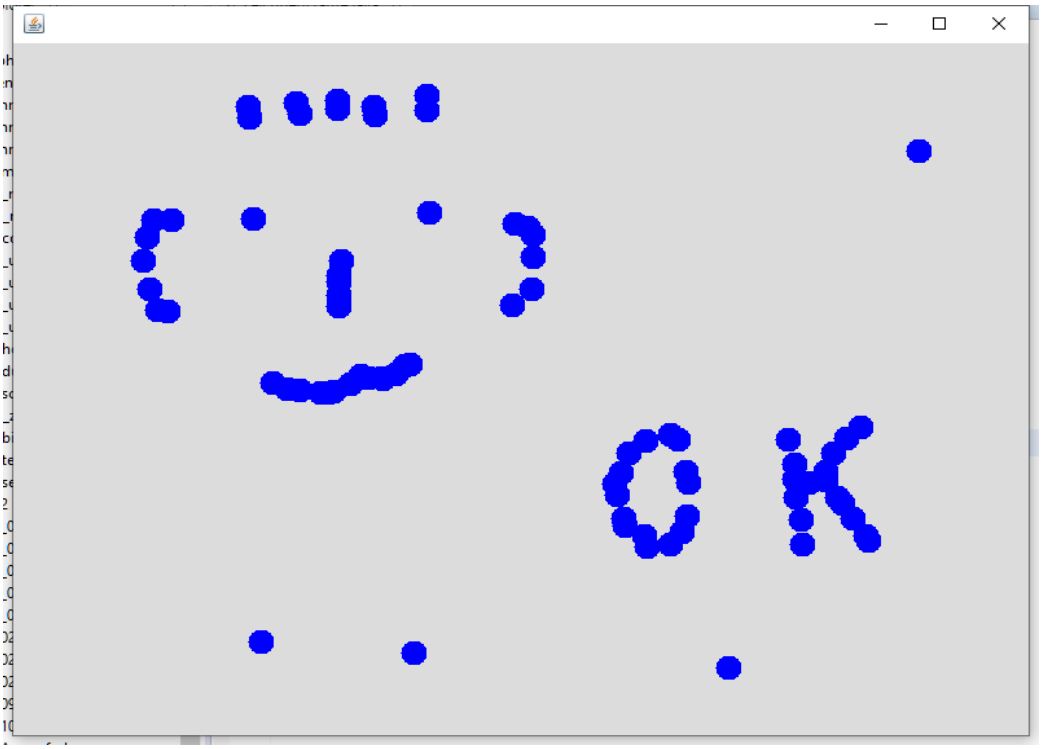
- paint Methode ändern
- durch klicken innerhalb des Fensters mit der LINKEN Maustaste soll sich die Farbe der linken Ellipse ändern
- dazu wichtige BEMERKUNG: das neuzeichnen eines Fensters kann von innen oder von aussen durch Aufruf der Methode **`repaint()`** erreicht werden!!!
- und denken Sie an das interface: **`MouseListener`**

Gruppenprojekt: simples Zeichenprogramm

Bitte erstellen Sie in der Gruppe ein einfaches „Zeichenprogramm“, mit dem

- innerhalb eines JFrame
- auf einem JPanel
- mit Mausklicks
- blaue Kreise gezeichnet werden können
- die das Panel sich dann auch merkt
- und bei jeder Aktualisierung auch wieder korrekt neu zeichnet

Das ganze soll (nach etwas Zeichnen mit Mausklicks) in etwa so aussehen:



Tipps zur Struktur der Software: gut geht das Ganze mit 4 Klasse,

- *ZeichnenApp* , die alles startet (wie im Skript)
- *ZeichnenFrame* , der in diesem Fall ein „dummes“ Fenster ist, das aber das
- *ZeichnenPanel* enthält, auf dem die Logik implementiert ist und das wiederum
- *Kreis* - Objekte zeichnet (die blauen Kreise) und in einem Vektor merkt

Die Idee ist dabei folgende – und wird hier nur in mittelfeinen Zügen dargestellt, damit Sie selber auch noch etwas Kreativität und Implementierungsarbeit reinstecken können.

Viel Spass und Erfolg beim Implementieren – nach dieser Übung sollten Sie schon etwas mehr an Erfahrung im Bereich „2D und Graphics“ besitzen.

- das *Panel* hat einen *Maus-Listener*, und zwar sich selber
- das *Panel* hat als Instanzvariable einen *Vektor*, in dem die gezeichneten *Kreise* gemerkt werden
- das Panel hat eine Methode *paint(Graphics gra)* mit der die bisher mit Maus geklickten *Kreise* alle gezeichnet werden
- der Kreis ist eine ganz einfache Klasse, in der einfach die x und y Koordinaten des Kreises als public (!!) Instanzvariablen gemerkt werden (((später mal können wir das eleganter implementieren…)))

Wenn jetzt mit der Maus auf das Panel geklickt wird,

- wird *mousePressed* aufgerufen
- dort wird die x und y Koordinate des Mausklicks festgestellt
- mit diesen Koordinaten wird ein *Kreis*-Objekt erstellt
- dieses *Kreis*-Objekt wird im *Vektor* gemerkt
- und mit *repaint*() wird ein Neuzeichnen angestossen

Die daraufhin aufgerufene Methode *paint(Graphics gra)* ist an sich ganz einfach:

- der bisherige *Panel*-Inhalt wird gelöscht
- es werden mit einer Schleife alle *Kreis*-Objekte des *Vektors* durchgelaufen
- jedes dieser Objekte wird als blaues Oval auf dem *Graphics* gezeichnet

Übungsaufgabe

Etwas schwerer:

Implementieren Sie eine Klasse **FrameMover** , deren Objekte

- jeweils einen Frame der Grösse 100x100 erstellen
- der jeweils eine andere, zufällige Hintergrundfarbe hat
- und auf dem Bildschirm darstellen
- und diesen Frame alle 3-5 Sekunden auf eine zufällige neue Stelle des Bildschirms bewegen (Klasse JFrame: setLocation(….))
- erstellen Sie 200 Objekte von **FrameMover** und starten Sie diese parallel!

Übungsaufgaben

Implementieren Sie folgende Funktionen/Methoden rekursiv:

A. zweiHochX(int x) : Funktion; zwei hoch x berechnen

B. aHochb(int a, int b): Funktion; a hoch b berechnen

C: fakultaet(int x) : Funktion; x! berechnen; (fakultät(5) ist bspw 1*2*3*4*5)

Etwas schwerer, da hier ggf „doppelte“ Rekursion verwendet werden muss (sprich: ZWEI Methoden, die beide rekursiv sind)

:

D. dreieckZeichnen(int x) : ein Dreieck aus o-Zeichen der Höhe x zeichnen.
(Beispiel zeichneDreieck(5) :)

```
o o o o o
o o o o
o o o
o o
o
```

E. zeichneDreieck(int x) : Dreieck aus Hash-Zeichen der Höhe x zeichnen. Andersrum.
(Beispiel zeichneDreieck(5) :)

```
#
# #
# # #
# # # #
# # # # #
```

Nebenbei: Aufgaben D und E funktionieren auch mit EINER rekursiven Methode, die dann aber ZWEI Parameter besitzt….. Ist aber nicht ganz leicht…..