PIS Klausur SS2014

A1) Threads

(6) a) Ein Thread kann verdrängt werden

1. in welchem Zustand ist er dann?

2. zu welchen Zeitpunkten in seiner run()-Methode kann dies geschehen?

3. Wer trifft die Entscheidung, welcher Thread verdrängt wird?

(2) b) Welcher Thread führt die main()-Methode aus?

(4) c) Thread.stop() ist gefährlich. Warum?

(4) d) Wie endet der Thread normalerweise?

(4) e) Thread.sleep(int), Object.wait() und andere können Ausnahme InterruptedException werfen. Unter welchen Umständen? Erläutere!

A2) Vererbung und Polymorphie

(3) a) Besitzt wirklich jede Klasse einen Konstruktor? Auch abstrakte Klasse, die ja nicht als Objektfabrik genutzt werden kann? Begründe!

(4) b) Erläutere den Unterschied zwischen statischen und dynamischen Binden! Wie Methoden, wie Attribute?

(5) c) Zähle alle Typen T auf, deren Variable eine Referenz auf ein Objekt der Klasse W speichern dürfen:

```
Class W extends Würfel implements Codec, Cloneable {…}

    T var = new W();

    // Welche Typen sind legal?
```

(9) d) Sei B eine von A abgeleitete Klasse, in der A.m() durch B.m() überschrieben wird:

```
Class A { …

        Public R m(P p) throws E { … }

}

Class B extends A { …

        Public R m(P p) throws F { … }

}
```

Erläutere mit Ersetzungsprinzip:

1. Wie hängt Klasseninvariante von A mit B zusammen?
2. Wie hängen Vor- und Nachbedingung von A.m() und B.m() zusammen?
3. Wie sollten Ausnahmen E & F zusammenhängen?

A3) Synchronisation

(4) a) 3 Klassen für Zeichenketten: String, StringBuffer, StringBuilder. Erläutere wesentliche Unterschiede!

(?) b) Sind Swing- Datenstrukturen threadsicher? Können Swing- Datenstrukturen inkonsistent werden? Erläutere!

(6) c)

```
public synchronized boolean istVoll() { … }

Public synchronized void einfuegen(Object object) {

        … // blockiert bei vollem Puffer

}
```

Erzeuger Thread soll auf keinen Fall blockiert warden und fragt vor dem Eintragen ab, ob er voll ist.

```
…

If(!p.istVoll() ){

        p.einfuegen(obj); // blockiert nie

} else { } // weitermachen, ohne einzufügen
```

Was hältst du von dieser Lösung?

A4) Grafische Benutzeroberfläche

(6) a) 1. Aufgabe des Layout Manager

2. Objekte vom Layout Manager

3. Welches Entwurfsmuster steckt dahinter?

(5) b) MVC-Architektur bei Swing-Komponenten

Wie lauten Anteile MVC in Komponente in JList? Welcher Anteil übernimmt Rolle des Beobachters im Beobachter-Entwurfsmuster?

(12) c) Methode EventHAndler. actionPerformed(…) wird auch im restlichen Programm nie aufgerufen. Dennoch wird sie ausgeführt, wenn der Benutzer die JButton-Objekt-Taste drückt.

1. Wie kann das sein?
2. In welchem Thread wird Methode ausgeführt?
3. Mit welchem Methodenaufruf findet man den Thread heraus?
4. Erläutere alle Schritte vom Eintritt Benutzerereignis Tastendruck bis zur Ausführung der Methode actionPerformed()
5. Welches Verhalten erwartet von Benutzeroberfläche, wenn Taste gedrückt wird?

A6) Gegeben ist threadsichere Klasse BlockierendeMailbox

```
Interface Mailbox{
    // Anweisungen
    Public void senden(String n) throws InterruptedException;
    // Anweisungen
    Public String empfangen() throws InterruptedException;
```

(12) a) Schreibe ein Test-Programm

2 Threadgruppen Sender und Empfänger mit jeweils 10 Threads angelegt

Threadgruppe Sender soll alle 10ms Nachrichten (String) in Mailbox senden.

Thread der Threadgruppe Empfänger soll alle 20ms Nachricht von Mailbox empfangen.

Behandlung InterruptedException nicht vergessen.

(6) b) Methode Mailbox.senden() soll InterruptedException werfen, wenn ausführender Thread Unterbrechungsanforderung mittels Interrupt erhält.

Schreibe Code- Fragment.

(4) c) Welchen Nachteil hat Verwendung der intrinsischen Sperre (synchronized, wait(), notify) gegenüber Sperre java.util.concurrent.locks.ReentrautLock?