Pis Ws 2019/2020

Aufgabe 1 : 21 punkte

Aufgabe 2 : 21 punkte

Aufgabe 3 : 20 punkte

Aufgabe 4 : 20 punkte

Aufgabe 5 : 30 punkte

Summe : 112

Aufgabe 1 (Threads)

(a)Ein Thread kann vom Scheduler verdrängt werden ?

1.Welchen Zustand nimmt er dann ein ? (2)

2.wie Können sie verhindern , dass ein Zustand verdrängt wird ? Erläutern Sie (3)

(b)Welcher Thread führt normalerweise eine actionPerformed (ActionEvent)-Methode (bzw handle (T event)) aus ? ist dies der gleiche Thread , der die main ()- Methode ausführt ? (2+1)

-> Der event dispatcher ist zuständig dafür

-> bei der main methode handelt es sich um den Thread „main “

- [3 Punkte]

(c) Die methode Thread.sleep(int) hat eine InterruptedException georfen was ist vermutlich geschehen ? (4)

-> Ein Thread wurde von einem anderenThread unterbrochen

- [2 punkte]

(d) was bewirkt die Anweisung klein.interrupt() ? (3)

(e)

1.was bewirkt das SwingUtilities.invokeLater (Runnable) (bzw Platform.runLater(Runnable)) ? (3)

2.Wenn sollten Sie diese methode verwenden , wenn nicht ? (3)

Aufgabe 2 (Vererbung , Polymorphie)

(a) Besitzt wirklich jeder klasse einen Konstruktor ? Auch eine abstrakte Klasse , die ja nicht als ObjektFabrik genutzt werden kann ? Erläutern und begründen sie ihre Antworten mit Hilfe der begriffe Konstruktorverkettung und KlassenInvariante ! (1+3)

->jeder klasse besitzt ein Konstruktor

(b)Methoden werden oft dynamish gebunden , also zur Laufzeit . Wieso macht Vererbung das erfordelich ? (4)

(c)zählen Sie alle typen T auf , für die die unterstehende Anweisung Korrekt ist :

Class W extends Würfel implements Codec , Cloneable {……

}

T var = new W() ; // welches Typen T sind erlaubt

(5)

->- T var = new W() ;

-Cloneable var 2 = new W ();

-Würfel var 0 = new W () :

-Codec var 1 = new W () ;

(d) Sei B eine von A abgeleitete Klasse , in der die methode 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 (….)

Erläutern Sie mit Hilfe des „ErsetzungsPrinzips“ Ihre Antworten auf Folgende Fragen:

1.Wie hängt die Klasseninvariante von A mit der von B zusammen ?

2.Wie hängen Vor- und Nachbedingung von A.m() und B.m() zusammen ?

3.Wie Sollten die Ausnahmen E und F zussamenhängen ?

(9)

->

1- B enthält die klassen invariante von A

2-B muss die Vorbedingungen von A nicht verschärfen und Nachbedingungen nicht abschwächen

3-da B eine checkedException ist , muss F eine Spezialisierung davon sein

- [6 Punkte] “(3 punkte Abzug wegen des Ersetzungsprinzips) „

Aufgabe 3 (Synchronisation)

(a) In der Java Klassen bibliothek gibt es drei klassen für zeichenketten: „String , StringBuffer, StringBuilder“ , erläutern Sie die wesentliche unterschiede zwichen diesen drei klassen ! (6)

-> -String ist Threadsicher und Unveränderlich

-StringBuilder ist veränderlich aber Threadsicher

-StringBuffer ist Threadsicher und veränderlich

- [6 punkte]

(b) Ein Thread t führt die Methode C.m2() aus , von der m1() aufgerufen wird. Führt dies zu einer erneuten Blockade ? erläutern Sie genau

```
Class C {
  ……
sychronized public void m1() {…}
          sychronized public void m2() {…
m1();
}
}
```

(6)

->es gibt kein blockade dank dem „Synchronized“ .. damit kann jedesmal nur ein Thread einzigen im kritischen Abschnitt zugreiffen

- [2 punkte]

(c) betrachten Sie das erzeuger – verbraucher-problem auf einem Puffer P , der u.a die Methoden

```
Public synchronized boolean istvoll () {…
 }
Public synchronized void einfügen (Object ob ) {
…//blockiert nur bei vollem Puffer
}
```

Besitz . Ein Erzeuger-Thread möchte auf keinen Fall blockiert werden und fragt daher vor dem Eintragen in der Puffer ab , ob dieser nicht voll ist :

```
…..
If (!p.istVoll() ) {
p.einfügen (obj) ; //blockiert nie , weil p nicht voll
} else () //weitermachen , ohne einzufügen
```

Was halten Sie von dieser Lösung ? begründen Sie ihre Bewertung genau !

(8)

Aufgabe 4 (Graphische Benutzeroberfläche)

(a) erläutern sie, was ein BorderLayout in einem container bewirkt ! (4)

-> ordnet die Komponents des containers in 5 Regionen

- [3 punkte]

(b) Erläutern sie das beobachter Entwurfsmuster ! geben Sie ein beispiel für eine KlassenBibliothek in der diese Entwurfsmuster eingesetzt wird !

(4)

->präsentation,Darstellung , Benutzereingabe

- [1 punkt]

(c) In einer klasse Ui wird die die graphische Oberfläche einer anwendung festgelegt:

```
class UI{…
JButton taste = new JButton (“druck mich“);
EventHandler eH = new EventHandler() ;
void UI (){ …
taste.addActionListener(eH);
…
}
….
}
```

Eine andere klasse EventHandler legt fest , was bei eintritt des Ereignisses „Taste gedruckt“ geschehen soll

```
Class EventHandler implement ActionListener {….
Void actionPerformed(ActionEvent ae) {…
…
If( SwingUtilities.isEventDispatchThread() ) {
  Thread.sleep(5*1000);
…….
```

Die Methode EventHandler.actionPerformed wird auch im restlichen Programm nie aufgerufen .Dennoch wird sie ausgeführt , wenn ein Benutzer das JButton- Objekt taste drückt

1.wie kann das sein ?

2.in welchem Thread wird diese methode ausgeführt ?

3.erläutern sie genau (also ohne das wort automatisch o.ä.) alle schritte vom Eintritt des Benutzerereignisses tastendruck bis zur aüsführung der Methode actionPerformed(..)

4.wird die sleep(..) Anweisung tatsächlich ausgeführt ? warum ?

(12)

-> 1-Ein Action listener wurde bei der quelle registriert . nach dem Empfangen der Ereignisse wird der listener die geeignete Methoden ausführen .

2.in dem Edt

3.klick auf einem Button -> Ereigniss objekt wird automatisch erzeugt und in dem Event queue Eingefügt -> Action Listener emfängt die Ereignisse -> JVM wird die geeignete Methoden für jeden objekt im queue ausführen

[1 punkt]

[3 punkte]

[2 punkte]

[]

Aufgabe 5 (Analyse und Ergänzung eines Programms)

Die implementierung der untenstehenden klasse BlockierendeMailbox, soll ergänzt werden

```
class BlockierendeMailbox{

  private String[] ablage; // speichert die Nachrichten
  private int in;          // Stelle zum Einfügen,
  private int out;         // Stelle zum Entfernen
  private int n;           // Zahl der belegten Plätze
  private int N;           // Kapazität der Mailbox

  //Initialisiert eine Mailbox mit Kapazität kap
  BlockierndeMailbox (int kap){

  }

  // die älteste Nachricht aus einer nicht leeren Mailbox nehmen.
  // Blockiert, wenn die Mailbox leer ist.
  // Wirft InterruptedException, wenn Thread.interrupt() aufgerufen wird
  public String empfangen() throws InterruptedException{

  }
...
```

a) Wie lautet die Klasseninvariante für diese Klasse? (6)

b) Welchen Wert haben die Attribute, wenn die Mailbox leer ist? (2)

c) Wenn die Mailbox voll ist und `in == 3` gilt, welche Werte haben dann die anderen Attribute? (2)

n = 4 ; in =3; N=4;

d) Vervollständigen Sie den Konstruktor (im obigen Fragment)! (4)

e) Vervollständigen Sie die Methode `empfangen` (im obigen Fragment)! (8)

f) Gilt die Klasseninvariante zu jedem Zeitpunkt während der Ausführung der Methode `empfangen`? Begründen Sie! (4)

g) Wenn die Klasseninvariante eingehalten wird, werden dann auch die Nachrichten in der richtigen Reihenfolge empfangen? Begründen Sie! (4)