Aufgabe 29

Die Teilaufgaben a), b) und c) wurden in Java implementiert und liegen in der Klasse ***Server.java*** dieser Abgabe bei

d)

Wechselseitige Ausschluss:

In der Methode ***daten_ablegen()*** ist entscheidend, dass sowohl kein Sicherungswunsch besteht, als auch die Anzahl der aktuellen Clienten kleiner oder gleich der Anzahl der Maximalen Clienten ist. Sobald eine der Bedingungen nicht erfüllt wird, wird der betroffene Client in eine Warteschleife geschickt, und wartet dort auf einen ***notifyAll()*** Aufruf.

Sicherungswunsch	AnzClient>MaxClients	A oder B
True	True	True
True	False	True
False	True	True
False	False	False

Anhand der Wahrheitstafel erkennt man bereits, dass die Bedingungen mit einem ODER verknüpft werden. Ein UND wäre hier kritisch, da es nur den Clienten in die Warteschleife schicken würde, wenn beide Bedingungen gleichzeitig erfüllt worden wären.

```
while (sicherungswunsch || (maxClients <= anzahlClients)) {
        try {
                wait();
        } catch (InterruptedException e) {      }
}
```

Klasse ***Server.java***, Methode ***daten_ablegen(Client c)***, Zeile 16ff

Eine weitere kritische Stelle befindet sich in der Methode ***sicherungAktivieren().*** Hierbei muss das Programm darauf achten, dass die Sicherung erst aktiviert wird, wenn keine weiteren Clients mehr Daten auf den Server abgelegen. Die Anzahl der Clients, die aktuell Daten ablegen, wird in der Klassenvariable ***anzahlClients*** gesichert und um eins erhöht, wenn ein Client Daten ablegt, oder um eins verringert, wenn die Daten abgelegt und der Vorgang beendet wurde.

In der Methode wird jetzt das Aktivieren der Sicherung Mithilfe des ***wait()*** Befehls in einer while-Schleife hinausgezögert. Die while-Schleife kontrolliert hierbei, ob die Anzahl der Clienten bei 0 ist, also wenn alle offenen Abgabe-Prozesse abgearbeitet wurden. (Vgl. Server.java, Z. 38-44)

Erst dann wird die boolsche Variable ***sicherungswunsch*** auf **true** gesetzt und alle wartenden Threads über den ***notifyAll()*** Befehl informiert. (Vgl. Server.java, Z. 46,47)

Dadurch nimmt der Server keine neuen Daten mehr an, wird aber dennoch über die wartenden Clienten informiert. (Vgl. Server.java, Z. 16-22)

Nach Beendigung der Sicherung wird der ***sicherungswunsch*** wieder auf **false** gesetzt und alle wartenden Threads über ***notifyAll()*** wieder geweckt.

Aufgabe 30)

Aufgabe	Lösung
a)	iv)
b)	ii)
c)	iii)
d)	ii)
e)	iv)