

Übungsblatt 9

IMPERATIVE PROGRAMMIERUNG und ALGEBRAISCHE GRUNDLAGEN

Abgabe: bis Donnerstag 23.12.2004, 13:00 Uhr in die Einwurfkästen im Untergeschoß des neuen Informatikgebäudes am Fasanengarten

**Erreichbare Punkte: 25 P / 35 T
(Praktische Aufgaben / Theoretische Aufgaben)**

Hinweis:

Alle Programmieraufgaben sind unter Verwendung der in der Vorlesung vorgestellten In-/Out-Klassen zu erstellen und (ähnlich wie die Beispielprogramme im Skript) in ausführlicher Form zu kommentieren.

Versehen Sie die Programme mit Eingabeüberprüfungen, die sicherstellen, dass der Anwender korrekte Daten eingegeben hat.

1 ZUSAMMENGESetzte ANWEISUNGEN

1.1 Fibonacci (5P)

Leonardo von Pisa alias Fibonacci erfand im 12. Jahrhundert die nach ihm benannten Zahlenfolge, die folgendermaßen definiert ist:

$$F(0)=0, F(1)=1, F(i)=F(i-2)+F(i-1)$$

Schreiben Sie ein Java-Programm, das für jede natürliche Zahl $n > 0$, die durch den Benutzer eingegeben wird, die n -te Fibonaccizahl $F(n)$ berechnet. Verwenden Sie hierfür neben arithmetischen Operationen und Zuweisungen nur eine for-Schleife. Die Berechnung soll **iterativ, nicht rekursiv** in der Methode `int fib(int)` erfolgen.

2 DATENSTRUKTUREN

2.1 Vor- und Nachteile *Garbage Collection* (4T)

Nennen Sie je zwei Vor- und Nachteile der automatische Freigabe und Bereinigung des Speichers, der so genannten *Garbage Collection*.

2.2 Lauflängencodierung (10P, 2T)

Die Lauflängencodierung (*run length encoding*) ist eine Komprimierungstechnik, bei der jede Zeichenfolge, die aus mehr als 2 gleichen Zeichen besteht, durch das Zeichen und die Länge der Folge dieses Zeichens codiert wird. Die Eingabe `ABBCCCKKKKKKK` wird zum Beispiel zu `ABBC3K7`.

- Schreiben Sie eine Methode, die einen String, der nur aus Buchstaben besteht, nach diesem Verfahren codiert. (5P)
- Schreiben Sie eine Methode, die einen nach diesem Verfahren codierten String decodiert. (5P)
- Ist es sinnvoll, eine komprimierte Zeichenfolge mit der Methode aus a) nochmals zu komprimieren? (2T)

Schreiben Sie ein Java-Programm `RleString`, das die beiden Methoden aus a) und b) nutzt. Der Benutzer soll auswählen können, ob er einen Eingabestring codieren, oder decodieren möchte. Danach soll die Eingabe des zu codierenden/decodierenden String erfolgen mit der anschließender Ausgabe des Ergebnisses. Stellen Sie bei der Eingabe des zu codierenden Strings sicher, dass der Anwender nur Buchstaben (a-z, A-Z) eingibt. Beim zu decodierenden String dürfen neben den Buchstaben auch Ziffern eingegeben werden. Das erste Zeichen muss aber in jedem Fall ein Buchstabe sein.

2.3 Transpositions-Chiffren (10P)

Die Transposition von Buchstaben kann zur Chiffrierung eines Textes eingesetzt werden. So genannte Transpositions-Chiffren ordnen die Buchstaben eines Klartextes dabei nach einem Schema oder einer geometrischen Figur um, etwa nach einer zweidimensionalen Matrix. Bei der Spaltentransposition wird der zu chiffrierende Text zeilenweise z.B. in eine Matrix eingelesen und danach spaltenweise entsprechend einer vorgegebenen Reihenfolge ausgelesen.

Beispielausgabe für eine 2*3 Matrix mit Ausgabe-Reihenfolge 3-2-1:

Zu chiffrierender Text : Informatik
Chiffrierter Text : fmnrlotak

Die Matrizen sehen in diesem Fall folgendermaßen aus:

	1	2	3
1	l	n	f
2	o	r	m

	1	2	3
1	a	t	i
2	k		

Beachten Sie: Nicht besetzte Stellen der Matrix werden beim Auslesen der Spalten übergangen!

Schreiben Sie ein Java-Programm, das eine Textzeile einliest und diese unter Verwendung einer 4*3-Matrix durch Spaltentransposition chiffriert und ausgibt. Die Spalten der Matrix sollen danach in einer im Programm fest definierten Reihenfolge 3-1-2 ausgegeben werden. Verwenden Sie dabei eine eigene Methode `cipher()`. Diese soll aus einem übergebenen Text und einer ebenfalls übergebenden Ausgabenreihenfolge der Spalten den chiffrierten Text erzeugt.

3 RELATIONEN

3.1 Nutzung von Relationen in der Informatik (5T)

- Nennen Sie drei Bereiche, in denen Relationen in der Informatik von Bedeutung sind. (3T)
- In welchem Zusammenhang ist die algebraische Struktur der Monoide in der Informatik von großer Bedeutung? (2T)

3.2 Nachweis: Halbgruppe – Monoid (15T)

Überprüfen Sie, ob es sich bei folgenden Strukturen um Halbgruppen oder Monoide oder keines von beidem handelt. Beweisen Sie Ihre Aussage. Sie dürfen zum Beweisen auf schon bekannte Eigenschaften aus der Vorlesung zurückgreifen. Zum Widerlegen genügen begründete Gegenbeispiele.

- Die Addition $+$ auf der Menge \mathbb{N}_0 , der natürlichen Zahlen mit der Zahl Null. (3T)
- Die Multiplikation $*$ auf der Menge $M = \{r \in \mathbb{R} \mid r > 1\}$, wobei \mathbb{R} die Menge der reellen Zahlen ist. (3T)
- Die Vektoraddition mit der Verknüpfung $+$: $(a, b) + (c, d) = (a + c, b + d)$ auf der Menge \mathbb{N}_0 , der natürlichen Zahlen mit der Zahl Null. (3T)
- Die Linksidentität mit der Verknüpfung $\#$: $a \# b = a$ auf der Menge \mathbb{N}_0 der natürlichen Zahlen mit der Zahl Null. (3T)
- Die Konkatenation von Zeichenketten (Strings) in Java. (3T)

3.3 Graphen und Netze (9T)

Gegeben ist der folgende Netzausschnitt. Es werden vier Endsysteme (1,5,6,9) mit sechs Routern (0,2,3,4,7,8) teilweise vernetzt (siehe Abbildung). Die Pfeile legen die Verbindungen zwischen den Komponenten sowie die Kommunikationsrichtung für die zu übertragenden Datenpakete fest.

Behalten Sie zur Bearbeitung der Teilaufgaben die Nummerierung bei.

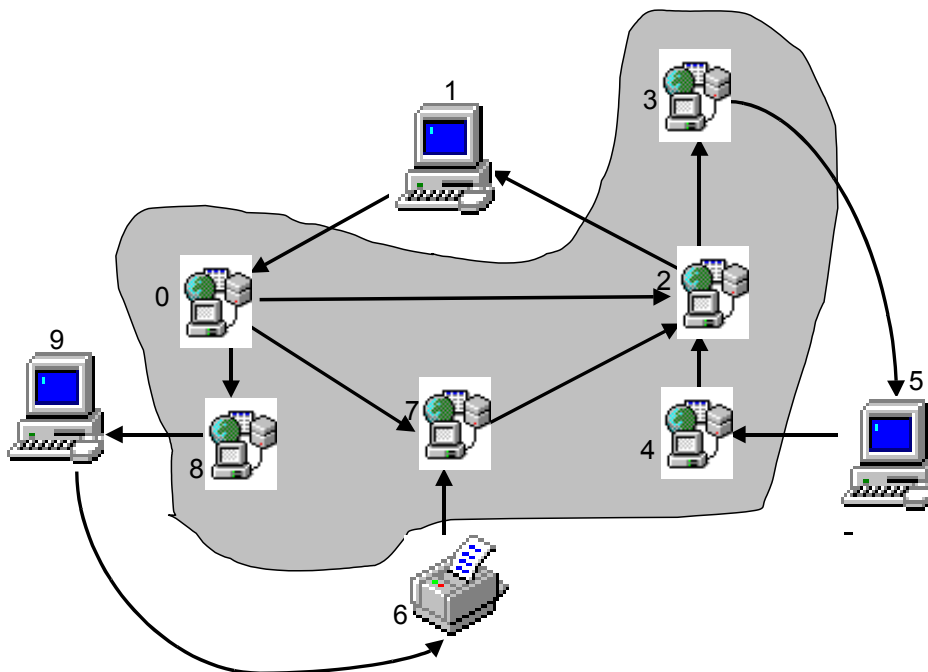


Abbildung zu Aufgabe Graphen und Netze

- Stellen Sie das Kommunikationsnetz als Graph dar. (2T)
- Was würde sich an der Aussage des Graphen ändern, wenn er ungerichtet wäre? (2T)
- Geben Sie für den gerichteten Graphen den Ein- und Ausgangsgrad jeder Ecke und den Grad des gesamten Graphen an. (5T)

Das Eulenfest:

Alles unter dem Motto:

Abtauchen

und

Auftauen

DIE PARTY UNTER DEM EIS

FACHSCHAFT INFORMATIK
FACHSCHAFT MATHEMATIK

Wann : Am 21.12.04, ab 20.00

Wo : Informatik Gebäude (50.34)

**Eingeladen sind alle, die kurz vor Weihnachten
noch mal richtig Party machen wollen**

**Seid bereit für eine heiße und coole Party!
(Live Musik, Happy Hour und Getränke passend zum Motto)**