FREISTAAT THÜRINGEN
Kultusministerium

ABITURPRÜFUNG 2009

LEISTUNGSFACH

INFORMATIK

(HAUPTTERMIN)

Bearbeitungszeit: 270 Minuten

Hilfsmittel: Wörterbuch zur deutschen Rechtschreibung
Taschenrechner (nicht programmierbar, nicht grafikfähig)
(Schüler, die einen CAS-Taschencomputer im Unterricht benutzen, dürfen diesen verwenden.)
Tafelwerk
PC mit Prolog-System und
Oberon-, Pascal- oder Java-System
Zufallszahlengenerator

Lösen Sie die Aufgaben 1 und 2 und wählen Sie von den Aufgaben 3.1 und 3.2 eine Aufgabe zur Bearbeitung aus.

Rechts unten neben jeder Teilaufgabe steht die für diese Teilaufgabe maximal erreichbare Anzahl von Bewertungseinheiten (BE).

Der Prüfungsteilnehmer sichert bei der praktischen Arbeit am PC mindestens alle 10 Minuten die von ihm erarbeiteten Quelltexte. Er hat die von ihm erarbeiteten Programme und Module im Quelltext zu kommentieren.
Die Quelltexte sind zusammen mit der Abiturarbeit abzugeben.

ÖFFNUNG AM 29. APRIL 2009

Aufgabe 1

1.1 Das Alphabet einer Sprache hat die fünf Buchstaben A, D, M, T und U. Wörter dieser Sprache sind unter anderem TAU, DATA, AM, DAMM, MUT und ATTA.

Geben Sie einen Binärcode an, der jedem Buchstaben der Sprache ein Codewort zuordnet. Alle Codewörter sollen gleich lang sein. Mit dem Binärcode sollen die Buchstaben des Alphabets der Sprache eindeutig codierbar sein.
Codieren Sie das Wort TUMDADA mit dem von Ihnen angegebenen Binärcode.
Entscheiden Sie, ob das von Ihnen codierte Wort eindeutig decodiert werden kann. Begründen Sie Ihre Entscheidung.

1.2 In Texten der Sprache wurden die Buchstaben ausgezählt. Daraus ergaben sich die in der Tabelle angegebenen Häufigkeiten.

A	D	M	T	U
33%	5%	20%	30%	12%

Geben Sie einen Binärcode an, der jedem Buchstaben der Sprache ein Codewort zuordnet. Das Codewort soll umso kürzer sein, je häufiger der Buchstabe auftritt. Mit dem Binärcode sollen die Buchstaben des Alphabets der Sprache eindeutig codierbar sein.
Codieren Sie das Wort TUMDADA mit dem von Ihnen angegebenen Binärcode.
Entscheiden Sie, ob das von Ihnen codierte Wort eindeutig decodiert werden kann. Begründen Sie Ihre Entscheidung.

1.3 Für E-Mail-Adressen in der Sprache aus Teilaufgabe 1.1 werden außer den fünf Buchstaben noch die Zeichen **@** und **.** verwendet.

Geben Sie eine Syntax für E-Mail-Adressen der Sprache an.
Stellen Sie Ihre Syntax in der Erweiterten Backus-Naur-Form (EBNF) oder als Syntaxdiagramm dar.
Geben Sie zwei E-Mail-Adressen an, die entsprechend Ihrer Syntax korrekt sind.
Ist ADAM@MUT.UTA@DM eine korrekte E-Mail-Adresse nach Ihrer Syntax? Begründen Sie Ihre Antwort.

1.4 Bei einem monoalphabetischen Verschlüsselungsverfahren wird ein Buchstabe eines Wortes durch immer den gleichen anderen Buchstaben ersetzt (Schlüssel).

Beispiel:

Buchstabe	A	D	M	T	U
Wird ersetzt durch:	T	U	A	D	M

Wort	DAMM	ATTA
Verschlüsselt:	UTAA	TDDT

Die Wörter ATDD, MDT, MTM, DU, ATMATM, UT und TAD sind Geheimwörter, die aus Wörtern der Sprache aus Teilaufgabe 1.1 nach dieser Methode erzeugt wurden.
Erläutern Sie, ob die Geheimwörter ohne Kenntnis des Schlüssels entschlüsselt werden können.
Bewerten Sie die Sicherheit des Verschlüsselungsverfahrens.

Setzen Sie sich mit der These auseinander, dass jede Verschlüsselung, die ein Mensch ersonnen hat, ein Mensch auch wieder entschlüsseln kann.

15 BE

Aufgabe 2

Gegeben sind die folgenden sechs Dominosteine d1 bis d6 (Abbildung 1). Jeder Dominostein ist in zwei Felder eingeteilt.

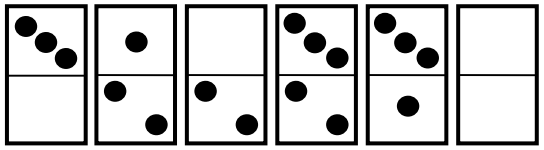


Abbildung 1

- Der Dominostein d1 enthält in einem Feld drei Punkte und im anderen Feld keinen Punkt.
- Der Dominostein d2 enthält in einem Feld einen Punkt und im anderen Feld zwei Punkte.
- Der Dominostein d3 enthält in einem Feld keinen Punkt und im anderen Feld zwei Punkte.
- Der Dominostein d4 enthält in einem Feld drei Punkte und im anderen Feld zwei Punkte.
- Der Dominostein d5 enthält in einem Feld drei Punkte und im anderen Feld einen Punkt.
- Der Dominostein d6 enthält in beiden Feldern keinen Punkt.

Die sechs Dominosteine sollen in einer Reihe nach folgenden Regeln angeordnet werden:

- Die Dominosteine dürfen sich nur an den kurzen Seiten berühren.
- Die Dominosteine dürfen nur mit Feldern gleicher Punktzahlen aneinander gereiht werden (Abbildung 2).

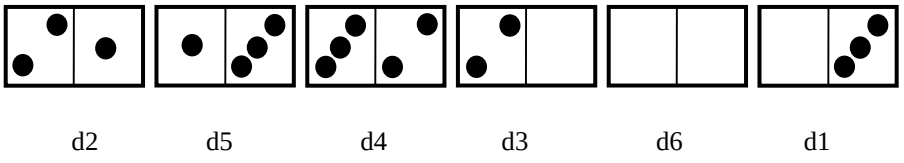


Abbildung 2

Geben Sie eine weitere regelgerechte Aneinanderreihung der sechs Dominosteine an.

Entwerfen und implementieren Sie ein Programm in Prolog, das alle regelgerechten Aneinanderreihungen der sechs Dominosteine ermittelt und ausgibt.

Geben Sie eine Anfrage in Prolog an, so dass alle regelgerechten Aneinanderreihungen der sechs Dominosteine ermittelt und ausgegeben werden.

15 BE

Aufgabe 3.1

Definieren Sie den Begriff binärer Baum.

Gegeben ist der in Abbildung 3 dargestellte binäre Baum. Begründen Sie, dass der Baum ein Suchbaum ist.

Geben Sie die Zahlen des Suchbaums in der Reihenfolge an, in der sie bei einem Inorder-Durchlauf ausgegeben werden.

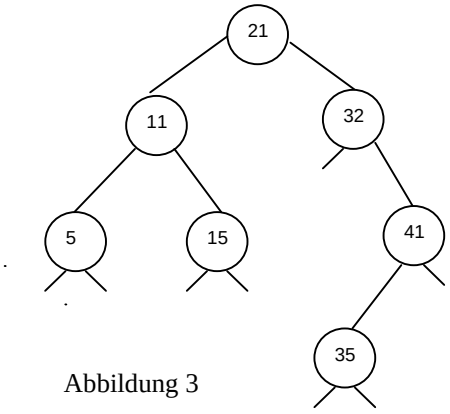


Abbildung 3

Die folgenden Wörter sollen in der gegebenen Reihenfolge in einen anfangs leeren Suchbaum eingefügt werden:
`Liste`, `Verbund`, `Objekt`, `Daten`, `Baum`, `Struktur`, `Folge`, `Matrix`, `Wiederholung`, `Entwurf`, `Graph`
Stellen Sie den entstehenden Suchbaum grafisch dar.

Die folgenden Wörter sollen in der gegebenen Reihenfolge in einen anfangs leeren Suchbaum eingefügt werden:
`Zuse`, `Wirth`, `Turing`, `Shannon`, `Ries`, `Leibniz`, `Hilbert`, `Euler`, `Church`, `Babbage`, `Ada`
Stellen Sie den entstehenden Suchbaum grafisch dar.

Vergleichen Sie die beiden Suchbäume hinsichtlich der Effizienz beim Suchen eines Wortes.

Entwerfen und implementieren Sie ein Modul, das einen Suchbaum mit den folgenden Operationen realisiert:

Operation	Beschreibung
erzeugen	Der leere Suchbaum wird erzeugt.
leer	Geprüft wird, ob der Suchbaum leer ist.
einfuegen(e)	Falls der Suchbaum existiert und der Wert e noch nicht im Suchbaum vorhanden ist, wird der Wert e in den Suchbaum eingefügt.
loeschen(e)	Falls der Suchbaum existiert und der Wert e im Suchbaum vorhanden ist, wird der Wert e aus dem Suchbaum gelöscht.
suchen(e)	Falls der Suchbaum existiert, wird geprüft, ob der Wert e in dem Suchbaum vorhanden ist.
inorderdurchlauf	Falls der Suchbaum existiert, werden die Werte des Suchbaums in Inorder-Reihenfolge ausgegeben.

Ein Wert ist dabei ein Wort.

Entwerfen und implementieren Sie ein Programm, das Ihr Modul importiert, verwendet und Folgendes leistet:

- Erzeugen des leeren Suchbaums
- Einfügen von Wörtern in den Suchbaum
- Löschen von Wörtern aus dem Suchbaum
- Suchen von Wörtern im Suchbaum
- Ausgeben der Wörter des Suchbaums

Testen Sie Ihr Programm durch folgende Handlungsreihenfolge:

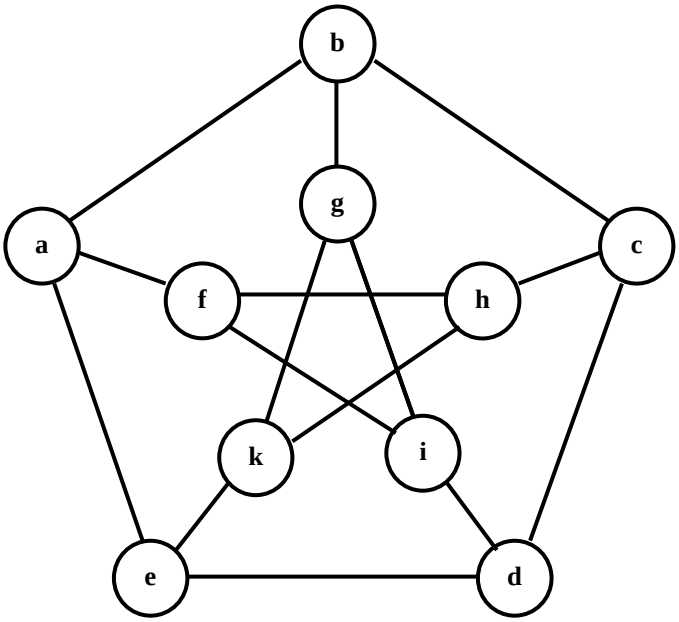
- Einfügen der Wörter `Liste`, `Verbund`, `Objekt`, `Daten`, `Baum`, `Struktur`, `Folge`, `Matrix`, `Wiederholung`, `Entwurf`, `Graph` in den Suchbaum
- Ausgabe der Wörter des Suchbaums
- Suchen der Wörter `Daten`, `Objekt` und `Prozedur` im Suchbaum
- Löschen der Wörter `Matrix`, `Struktur` und `Liste` in der gegebenen Reihenfolge aus dem Suchbaum
- Einfügen der Wörter `Shannon`, `Euler`, `Ada` und `Zuse`
- Ausgabe der Wörter des Suchbaums
- Suchen der Wörter `Zuse`, `Matrix`, `Struktur` und `Liste` im Suchbaum

Dokumentieren Sie den Test.

30 BE

Aufgabe 3.2

Der Petersen-Graph ist nach dem dänischen Mathematiker Julius Peter Christian Petersen (1839-1910) benannt. Der Graph besitzt die zehn Knoten a, b, c, d, e, f, g, h, i und k sowie fünfzehn ungerichtete Kanten.



Suchen Sie im Petersen-Graph einen Pfad, der jeden Knoten genau einmal enthält. Geben Sie diesen Pfad an.

Erläutern Sie am Beispiel der Suche nach einem Pfad im Petersen-Graph die Problemlösungsmethode Backtracking.

Der Petersen-Graph kann in einer Tabelle dargestellt werden.

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**k**
a	0	1	0	0	1	1	0	0	0	0
b	1	0	1	0	0	0	1	0	0	0
c	0	1	0	1	0	0	0	1	0	0
d	0	0	1	0	1	0	0	0	1	0
e	1	0	0	1	0	0	0	0	0	1
f	1	0	0	0	0	0	0	1	1	0
g	0	1	0	0	0	0	0	0	1	1
h	0	0	1	0	0	1	0	0	0	1
i	0	0	0	1	0	1	1	0	0	0
k	0	0	0	0	1	0	1	1	0	0

Die Tabelle gibt darüber Auskunft, welche Knoten im Petersen-Graph durch ungerichtete Kanten direkt miteinander verbunden sind.

Entwerfen und implementieren Sie ein Programm, das Folgendes leistet:

- Einlesen eines Startknotens
- Suchen nach einem Pfad im Petersen-Graph, der jeden Knoten genau einmal enthält
- Ausgabe des gefundenen Pfades

Testen Sie Ihr Programm. Dokumentieren Sie den Test.
Begründen Sie die Wahl Ihrer Datenstruktur zum Speichern des Graphen.

30 BE