FernUniversität in Hagen

Universitätsrechenzentrum

Bitte hier unbedingt Matrikelnummer und Adresse eintragen. Sonst ist keine Bearbeitung möglich.

Postanschrift: FernUniversität D-58084 Hagen

Name, Vorname

Straße, Nr.

PLZ, Wohnort

Deckblatt zur PASCAL-Klausur 1575 am 21.02.2004

Ihr Zeichen/Schreiben vom	Mein Zeichen/Auskunft erteilt	Telefon 02331 / 987-	Hagen,
	H. Dr. Wenzel / H. Kohl	2848 / 2815	21.2.2004

Bitte füllen Sie das **Adressfeld** sowie die Angaben zur **Matrikelnummer** deutlich lesbar aus. Ebenso ist die **Finanzamtsbescheinigung** auszufüllen, die Sie später unterschrieben zurück erhalten.
Nach Korrektur Ihrer Klausur erhalten Sie eine Kopie dieses Deckblattes zugeschickt; bei bestandener Klausur erhalten Sie anschließend mit getrennter Post nach einiger Zeit Ihren Schein.

Klausurdauer ist 3 (drei) Stunden.
Außer Papier und Schreibzeug sind keine Hilfsmittel bei der Klausur zugelassen.
Bitte schreiben Sie Ihre Lösungen in Reinschrift auf die dafür vorgesehenen nummerierten Seiten hinter der jeweiligen Aufgabenstellung.

Das Ergebnis Ihrer Klausur:

Nr. der Aufgabe	max. Punktzahl	erreichte Punkte
1	15	
2	25	
3	35	
4	25	
Insgesamt erreichte Punktzahl		
Notwendige Punktzahl		**50**

Die Klausur ist

☐ bestanden
☐ nicht bestanden

Datum _____ Unterschrift des Korrektors: _____

Telefonvermittlung 02331 / 987 -01
Telefax 02331/987-2720
Buslinien 515,514, 527,
E-Mail Hans.Wenzel@FernUni-Hagen.de

Aufgabe 1: Römische Zahlen

Wer kennt das Problem nicht: Sie stehen vor einer Inschrift, auf der eine Jahreszahl in römischer Schreibweise angegeben ist und rätseln, welches Jahr da wohl gemeint ist. Wie schön wäre es, in einer solchen Situation das Notebook aufschlagen und die Zahl in eine arabische Zahl umrechnen zu können. Das wollen wir in dieser Aufgabe realisieren.
Schreiben Sie ein Pascal-Programm, das eine Folge von römischen Zahlen aus der Standardeingabe einliest und den Wert als arabische Zahl auf die Standardausgabe ausgibt bis die Eingabe erschöpft ist.

Die römischen Zahlzeichen und ihr Wert sind:

M	1000
D	500
C	100
L	50
X	10
V	5
I	1

Die Zahlzeichen stehen in einer römischen Zahl in absteigender Wertigkeit, wobei die Wertigkeit der einzelnen Zahlzeichen zu addieren ist. Eine Ausnahme bilden dabei die Zeichen C, X und I, die auch vor einem höherwertigen Zeichen stehen können und dann zu subtrahieren sind. Es muß also stets das nächste Zeichen betrachtet werden, um entscheiden zu können, ob das aktuelle Zeichen zu addieren oder zu subtrahieren ist (s. folgende Beispiele).
Sie können bei dem Programm davon ausgehen, daß nur gültige römische Zahlen eingegeben werden; es ist also **keine Gültigkeitsprüfung** der Eingabe vorzunehmen. Es sollen aber Zeichen in der Eingabe ignoriert werden, die nicht als römische Zahl zu interpretieren sind.

Bei der Eingabe von

```
I
II
III
IV
V
VI
VII
VIII
IX
XI
XVIII
XIX
XX
XXVIII
XXIX
XL
XLIX
LIX
XC
```

```
XCIX
CL
CD
CM
CMXC
MCMXCIX
MM
Anno MMIV
```

sollte die Ausgabe etwa folgendermaßen aussehen:

```
I = 1
II = 2
III = 3
IV = 4
V = 5
VI = 6
VII = 7
VIII = 8
IX = 9
XI = 11
XVIII = 18
XIX = 19
XX = 20
XXVIII = 28
XXIX = 29
XL = 40
XLIX = 49
LIX = 59
XC = 90
XCIX = 99
CL = 150
CD = 400
CM = 900
CMXC = 990
MCMXCIX = 1999
MM = 2000
A ungueltige Eingabe ! Zeichen A wird ignoriert.
n ungueltige Eingabe ! Zeichen n wird ignoriert.
n ungueltige Eingabe ! Zeichen n wird ignoriert.
o ungueltige Eingabe ! Zeichen o wird ignoriert.
  ungueltige Eingabe ! Zeichen   wird ignoriert.
MMIV = 2004
```

Benotung: maximal 15 Punkte

Aufgabe 2: Magisches Quadrat

Ein magisches Quadrat ist eine quadratische Matrix, bei der alle Zeilensummen und alle Spaltensummen denselben Wert ergeben. Für ungerade natürliche Zahlen n gibt es einen Algorithmus, mit dem die ersten n^2 natürlichen Zahlen zu einem magischen Quadrat angeordnet werden können:

1. Die erste Zahl i=1 wird in die Mitte der ersten Zeile eingetragen.
2. Die Position der nächsten Zahl i+1 ergibt sich aus der Position der Zahl i folgendermaßen:
   (a) Ist i ein ganzzahliges Vielfaches von n, dann wird i+1 unmittelbar oberhalb von i eingetragen.
   (b) Anderenfalls direkt links unterhalb von i.
   (c) Der obere und untere Rand, sowie der linke und rechte Rand sind dabei zyklisch aneinander angeschlossen – miteinander verklebt, wie der Topologe sagt. Z.B. steht ein Element unmittelbar über einem Element der ersten Zeile, wenn es an der entsprechenden Stelle der letzten Zeile steht.

Schreiben Sie für dieses Problem ein Pascal-Programm, das die folgenden **Anforderungen** erfüllt:

- Die maximale Größe max=15 wird als Konstante definiert.
- Die Zahlen werden in einem zweidimensionalen ARRAY gespeichert.
- Für die Indexweiterschaltung sind zwei Funktionen *vor* und *nach* zur Berechnung des Vorgängers bzw. Nachfolgers des aktuellen Index bei der zyklischen Zählung modulo n zu schreiben.
- Es wird eine Zahl n über die Standardeingabe eingelesen, auf Zulässigkeit überprüft und bei Zulässigkeit das magische Quadrat mit der Kantenlänge n auf die Standardausgabe in tabellarischer Form ausgegeben. Die Feldlänge aller Zahlen ist dabei so zu wählen, daß die größte Zahl n^2 mit einer führenden Leerstelle ausgegeben wird.

Im folgenden die Beispiele für n=3, n=4 und n=15:

```
Hier ist das magische Quadrat.
Bitte geben Sie eine ungerade Zahl von 1 bis 15 an: 3
magisches Quadrat mit der Kantenlaenge 3

 6 1 8
 2 9 4
 7 5 3


Hier ist das magische Quadrat.
Bitte geben Sie eine ungerade Zahl von 1 bis 15 an: 4
Koennen Sie nicht lesen ? Das war's.
```

```
Hier ist das magische Quadrat.
Bitte geben Sie eine ungerade Zahl von 1 bis 15 an: 15
magisches Quadrat mit der Kantenlaenge 15

 120 103  86  69  52  35  18   1 224 207 190 173 156 139 122
 104  87  70  53  36  19   2 225 208 191 174 157 140 123 106
  88  71  54  37  20   3 211 209 192 175 158 141 124 107 105
  72  55  38  21   4 212 210 193 176 159 142 125 108  91  89
  56  39  22   5 213 196 194 177 160 143 126 109  92  90  73
  40  23   6 214 197 195 178 161 144 127 110  93  76  74  57
  24   7 215 198 181 179 162 145 128 111  94  77  75  58  41
   8 216 199 182 180 163 146 129 112  95  78  61  59  42  25
 217 200 183 166 164 147 130 113  96  79  62  60  43  26   9
 201 184 167 165 148 131 114  97  80  63  46  44  27  10 218
 185 168 151 149 132 115  98  81  64  47  45  28  11 219 202
 169 152 150 133 116  99  82  65  48  31  29  12 220 203 186
 153 136 134 117 100  83  66  49  32  30  13 221 204 187 170
 137 135 118 101  84  67  50  33  16  14 222 205 188 171 154
 121 119 102  85  68  51  34  17  15 223 206 189 172 155 138
```

Benotung: maximal 25 Punkte

Aufgabe 3: Wechselgeld

Wir wollen den gestreßten Kassiererinnen im Supermarkt die Arbeit weiter erleichtern. Moderne Registrierkasssen weisen wohl das herauszugebende Wechselgeld aus, aber wir wollen das weiter verbessern, so daß zusätzlich die herauszugebenden Banknoten und Münzen ausgewiesen werden, wobei die Anzahl der Einheiten minimiert wird, also möglichst große Einheiten gewählt werden. Als Prototyp soll ein Pascal-Programm die herauszugebenden Beträge als reelle Zahl zeilenweise aus der Standardeingabe lesen und die Stückelung über die Standardausgabe ausgeben.
Um das Programm international vermarkten zu können, sollen die Währungseinheitsbezeichnungen über Konstantendefinitionen festgelegt und die Werte der verfügbaren Banknoten und Münzen über die ersten beiden Eingabezeilen der Standardeingabe eingelesen werden, wobei in der ersten Zeile die Einheiten der Vorkomma- und in der zweiten Zeile die Einheiten der Nachkommastellen vorliegen.
Für das Euro–Europa beispielsweise wären die Einheiten **Euro** und **Cent** und die Eingabe hätte das folgende Aussehen:

```
500 200 100 50 20 10 5 2 1
50 20 10 5 2 1
0.01
999.99
123.84
```

Die Ausgabe sollte dann etwa folgendermaßen aussehen:

```
Rueckgeld :        0.01 Euro
0 Euro und 1 Cent
 1 x 1 Cent
Rueckgeld :      999.99 Euro
999 Euro und 99 Cent
 1 x 500 Euro
 2 x 200 Euro
 1 x 50 Euro
 2 x 20 Euro
 1 x 5 Euro
 2 x 2 Euro
 1 x 50 Cent
 2 x 20 Cent
 1 x 5 Cent
 2 x 2 Cent
Rueckgeld :      123.84 Euro
123 Euro und 84 Cent
 1 x 100 Euro
 1 x 20 Euro
 1 x 2 Euro
 1 x 1 Euro
 1 x 50 Cent
 1 x 20 Cent
 1 x 10 Cent
 2 x 2 Cent
```

Schreiben Sie für dieses Problem ein Pascal-Programm, das die folgenden **Anforderungen** erfüllt:

- Konstantendefinition für die Währungseinheitsbezeichnungen.
- Je eine verkettete Liste für die verfügbaren Vorkomma- und Nachkommaeinheiten, deren Elemente neben dem jeweiligen Wert auch die herauszugebende Anzahl aufnehmen können:

```
+--------+---+
| Wert   |   |  Naechster
+--------+ O-+----->
| Anzahl |   |
+--------+---+
```

- Prozedur zur Initialisierung der Listen gemäß den ersten beiden Eingabezeilen und darin
  - Funktion zum Generieren eines Knotens der Liste.
  - Funktion zum Generieren einer Liste, die sowohl für die Vor- (Euro), wie für die Nachkommastellen (Cent) benutzt werden kann.
- Prozedur zur Aufteilung einer ganzen Zahl auf die verfügbaren Einheiten, die sowohl für die Vor- (Euro) wie für die Nachkommastellen (Cent) benutzt werden kann.
- Prozedur zur Ausgabe der herauszugebenden Einheiten, die sowohl für die Vor- (Euro) wie für die Nachkommastellen (Cent) benutzt werden kann.

Das Programm hat die folgende Grobstruktur:

- Initialisierung
- Solange Daten vorhanden
  - Rückgabebetrag lesen
  - Rückgabebetrag in Vor- und Nachkommastellen aufteilen
  - Stückelung des Vorkommabetrags ermitteln
  - Stückelung des Nachkommabetrags ermitteln
  - Stückelung des Vorkommabetrags und Währungseinheit ausgeben
  - Stückelung des Nachkommabetrags und Währungseinheit ausgeben

Benotung: maximal 35 Punkte

Aufgabe 4: Kaninchenvermehrung

In dieser Aufgabe wollen wir das Wachstum einer Kaninchenpopulation behandeln. Dazu machen wir einige vereinfachende Annahmen:

- Zu Beginn des ersten Monats setzen wir ein neu geborenes Kaninchenpaar aus.
- Kaninchen werden nach 2 Monaten geschlechtsreif und haben eine Tragezeit von einem Monat, sind sofort wieder empfängnisbereit und haben in jedem Wurf ein Kaninchenpaar als Nachkommen. D.h. ab dem dritten Lebensmonat wird von jedem Kaninchenpaar monatlich ein Nachkommenpaar erzeugt.
- Kaninchen leben ein Jahr, haben also insgesamt 10 Nachkommenpaare.

Aus diesen Annahmen ergibt sich die Zahl $f(n)$ der Kaninchenpaare am Ende des n-ten Monats zu

$$f(n) = \begin{cases} 0 & , \quad n \leq 0 \\ 1 & , \quad n = 1 \\ f(n-1) + f(n-2) - f(n-12) & , \quad n \geq 2 \end{cases}$$

Schreiben Sie ein Pascal-Programm, das den gewünschten Monat einliest und die Zahl der Kaninchenpaare am Ende des gegebenen Monats ausgibt. Benutzen Sie dazu eine rekursive Funktion.
Die Ausgabe sollte etwa folgendermaßen aussehen:

```
Kaninchenpaare am Ende welchen Monats ? : 20
Am Ende des 20-ten Monats gibt es 6635 Kaninchenpaare.
```

Benotung: maximal 25 Punkte