Prüfziffernverfahren

Diplomarbeit

für den Studiengang
Betriebswirt (VWA) - Schwerpunkt Informationsmanagement

Eingereicht von:

Kurt Zelt
Wilhelmstr. 1
68723 Plankstadt
tel. 06202/924723
www.zelt-online.de

Prüfer:

Dipl.-Math. Edith Mechelke-Schwede
Viernheim

Verwaltungs- und Wirtschafts-Akademie
Rhein-Neckar e. V., Mannheim

Mannheim, September 2002

Inhalt

Inhalt ... 1
Vorwort ... 3
1 Notwendigkeit von Fehlerverhütung bei Nummern ... 5
1.1 Darstellung von Informationen ... 5
1.2 Auftreten von Fehlern bei Nummern ... 5
1.3 Fehlerarten ... 6
1.4 Fehlerquellen ... 7
1.5 Fehlervermeidung durch organisatorische Maßnahmen ... 8
1.6 Fehlererkennungsmethoden ... 10
1.6.1 Tabellenkontrolle ... 10
1.6.2 Kombination von Informationen ... 10
1.6.3 Prüfzeichen ... 11
1.6.4 Multiplikatorverfahren ... 12
2. Anforderungen an ein Prüfziffernverfahren ... 13
2.1 Stellenanzahl ... 14
2.2 Modulus-Wert ... 14
2.2.1 Modulus-11-Verfahren ... 15
2.2.2 Modulus-10-Verfahren ... 16
2.2.3 Modulus-7-Verfahren und Modulus-9-Verfahren ... 16
2.2.4 Modulus-23-Verfahren und Modulus-26-Verfahren ... 17
2.2.5 Modulus-97-Verfahren ... 17
2.2.6 Modulus-99-Verfahren ... 18
2.2.7 Modulus-101-Verfahren ... 18
2.2.8 Dreistellige Prüfziffernverfahren ... 18
2.3 Gewichtung der Ziffern ... 19
2.3.1 Modulus-11-Verfahren ohne Gewichtung ... 19
2.3.2 Modulus-10-Quersummenverfahren ... 20
2.3.3 Erkennen von Einzeichenfehlern ... 21
2.3.4 Erkennen von Vertauschungen ... 23
2.3.5 Erkennen von Bündelfehlern ... 23
2.4 Laufende Quersummenbildung ... 24
3 Arten von Prüfziffernverfahren ... 27
3.1 Die ISBN ... 27
3.2 Die EAN ... 28

3.3 Prüfziffern bei Bankkontonummern 30
3.3.1 Modulus 10 mit Gewichtung 2, 1 30
3.3.2 Modulus 10 mit laufender Quersummenbildung 31
3.3.3 Modulus 10 mit Gewichtung 3, 7, 1 32
3.3.4 Modulus 11 mit Gewichtung 2, 3, 4, 5, 6, 7, 8, 9, 10 33
3.3.5 Modulus 11 mit Gewichtung 2, 3, 4, ... 10 (modifiziert) 34
3.3.6 Modulus 11 mit Gewichtung 2, 3, 4, 5, 6, 7 (modifiziert) 34
3.4 Die IBAN 35
3.4.1 Aufbau der IBAN 36
3.4.2 Berechnung der Prüfziffer 37
3.4.3 Kontrolle der IBAN-Prüfziffer 39
4 Zusammenfassung 41
4.1 Allgemeines 41
4.2 Kritik an heutigen Verfahren 41
4.3 Welches Prüfziffernverfahren für welches Nummernsystem? 42
4.4 Ausblick 43
5 Programmbeschreibung 44
5.1 Allgemeines 44
5.2 Aufruf des Programms über die Schaltfläche 44
5.3 Aufruf der Funktionen über die Symbolleiste 46
5.4 Beschreibung des Programms 47
5.5 Fehlerbehandlung 49
Anhang 50
Quellenverzeichnis 81
Abbildungsverzeichnis 83
Eidesstattliche Versicherung 84

Vorwort

Wozu braucht man überhaupt eine Prüfziffer?

Bei der Datenerfassung mehrstelliger Nummern (Kontonummern, Artikelnummern, Versicherungsnummern usw.) werden heute in der Regel Prüfziffernverfahren verwendet, um eine möglichst fehlerfreie Verwaltung und Bearbeitung der Daten sicher zu stellen.

Die hier durchzuführende Analyse soll zuerst einen Überblick über die grundsätzliche Problematik der Verwendung von Ziffernfolgen bei der Datenverarbeitung schaffen. Erkenntnisse über Fehlerquellen und –ursachen haben zur Entwicklung immer komplexer werdender Prüfziffernverfahren geführt, die dann im Mittelpunkt dieser Arbeit stehen. Um das Verständnis für ihre Funktionsweise vertiefen zu können, werden mit ein paar Beispielen verschiedene Verfahren detaillierter beschrieben. Eine kritische Betrachtung rundet das Thema ab.

Selbstkorrigierende Nummern sind nicht Thema dieser Arbeit. Sie sind den Prüfziffern-verfahren von der Funktionsweise her sehr ähnlich, erreichen aber mit relativ hohem Aufwand leider nur mäßige Ergebnisse.

Im ersten Kapitel wird die Bedeutung von Nummern bei der Darstellung von Informationen untersucht. Die notwendige manuelle Eingabe numerischer Daten weist eine Vielzahl möglicher Fehlerquellen auf. Aber auch beim maschinellen Einlesen von Zahlen können Fehler entstehen. Um korrekte Ergebnisse zu gewährleisten, kommen spezifische organisatorische Maßnahmen zum Einsatz. Darüber hinaus haben sich Fehlererkennungsmethoden durchgesetzt. Die Tabellenkontrolle, die Kombination von Informationen, das Prüfzeichen und das Multiplikatorverfahren sind solche Methoden. Insbesondere die rechnerisch ermittelten Prüfzeichen und die Prüfziffernverfahren sorgen wegen ihrer hohen Fehlererkennungswahrscheinlichkeit für eine effiziente Datenverarbeitung.

Die verschiedenen Prüfziffernverfahren sind Gegenstand des zweiten Kapitels. Es gibt ein- und mehrstellige Prüfziffern, zu deren Berechnung verschiedene Modulus-Werte zugrunde gelegt werden können. Am Verbreitetsten sind die Modulus-10-Verfahren und die Modulus-11-Verfahren, die entsprechend ausführlich expliziert werden. Außerdem werden einige andere Alternativen aus der Literatur vorgestellt.

Die einfachen Prüfziffernverfahren weisen unter Umständen immer noch erhebliche Mängel bei der Fehlererkennung auf. Um die Sicherheit bei der Dateneingabe zu optimieren, werden innerhalb der jeweiligen Verfahren für die Berechnung der Prüfziffer zusätzlich Gewichtungsfaktoren (Stellenmultiplikatoren) verwendet. Zum Vergleich zeigen Rechenbeispiele mit einfachen und erweiterten Methoden die Vorteile der Berechnung mit Gewichten. Eine andere Möglichkeit zur exakteren Bestimmung

der Fehler stellt die laufende Quersummenbildung dar, die vor allem bei einigen Modulus-10-Verfahren vorkommt.

Gegenstand des dritten Kapitels sind in der Praxis vorkommende Nummernsysteme. Die Buchnummer (ISBN) verwendet ein Modulus-11-Verfahren mit den Gewichten 2 bis 10 in aufsteigender Reihenfolge. Bei der Artikelnummer (EAN) kommt das einfacher zu berechnende Modulus-10-Verfahren mit den sich wiederholenden Gewichten 3 und 1 zur Anwendung. Sparkassen und Banken verwenden für ihre Kontonummernsysteme entweder ein Modulus-10- oder ein Modulus-11-Verfahren. Hierbei kommen verschiedene Möglichkeiten in betracht, Gewichtungsfaktoren und Modifikationen einzusetzen. Die internationale Bankkontonummer (IBAN) hat bis zu 34 Stellen und enthält eine zweistellige Prüfziffer nach dem Modulus-97-Verfahren.

Das vierte Kapitel bietet eine kurze Zusammenfassung des Themas. Maßnahmen zur Fehlervermeidung bleiben oberstes Gebot. Zur Fehlererkennung in Nummernsystemen weisen Prüfziffernverfahren die besten Ergebnisse auf. Die verbreitetsten Verfahren könnten unter Umständen verbessert werden.

Teil dieser Arbeit ist ein Programm zur Errechnung und Prüfung von Prüfziffern. In Kapitel 5 sind Aufbau und Funktionsweise des Programms beschrieben. Der kommentierte Quellcode ist im Anhang abgedruckt. Außerdem liegt dieser Arbeit ein Datenträger bei.

1 Notwendigkeit von Fehlerverhütung bei Nummern

1.1 Darstellung von Informationen

Die maschinelle Datenverarbeitung erlaubt es, große Mengen von Informationen zu speichern. Um jedoch Informationen in einem Computer speichern zu können, müssen sie in ein für ihn verständliches Format umgewandelt werden, zum Beispiel in Form von Buchstaben und Zahlen. So werden aus Informationen über Objekte Daten.

"**Daten** (engl: data) stellen **Informationen** (das heißt Angaben über Sachverhalte und Vorgänge; engl.: information) aufgrund bekannter oder unterstellter Abmachungen in einer maschinell verarbeitbaren Form dar."[1]

Die hier erwähnten Abmachungen sind notwendig, damit die Daten hinterher in irgendeiner Form ausgewertet werden können. Sinn und Zweck der maschinellen Datenverarbeitung ist es schließlich, aus den gespeicherten Daten wiederum Informationen zu gewinnen.

Daten müssen so strukturiert sein, dass ein schneller Zugriff, eine leichte Aktualisierung und eine beliebige Auswertung möglich sind.[2]

Um dies zu gewährleisten, werden Ordnungsbegriffe verwendet. In der betrieblichen Praxis werden alle Datenobjekte wie Kunden, Mitarbeiter, Artikel, Konten, Kostenstellen usw. mit einem künstlichen Attribut versehen, der Objektnummer.[3]

Eine Objektnummer ist nicht zwangsläufig numerisch, kann also auch alphabetische Zeichen enthalten. Umgangssprachlich wird hierfür der Begriff *Schlüssel* benutzt. In Anlehnung an die Norm *DIN 6763 (Nummerung)* werden im Folgenden die Begriffe *Nummer* und *Benummern* verwendet.

1.2 Auftreten von Fehlern bei Nummern

Die hohe Verbreitung maschineller Datenverarbeitung hat dazu geführt, dass auch die Bedeutung der Benummerung von Objekten wie Kunden, Artikel usw. zugenommen hat. Die Nummern sind der Ordnungsbegriff und der Schlüssel zu den Informationen.

---

1 Hansen, Hans Robert, Wirtschaftsinformatik I, Grundlagen betrieblicher Informationsverarbeitung, 7. Aufl., Stuttgart 1998, S. 6

2 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, Einführung in die Wirtschaftsinformatik, 10. Aufl., Berlin und Heidelberg 2002, S. 135

3 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 146

Aufgrund der *Identnummer* werden die Informationen zur Verarbeitung oder Anzeige bereitgestellt.[4] Da Nummern weniger bezeichnend als die genauen Beschreibungen sind, besteht jedoch die Gefahr von Verwechslungen.

Entsprechend führt eine falsche Identnummer zu den falschen Objektdaten und man erhält fehlerhafte Verarbeitungsergebnisse.[5] Eine falsche Materialnummer zieht eine falsche Ausgangsbuchung nach sich, eine falsche Personalnummer eine falsche Lohnzahlung und eine falsche Bankkontonummer führt zur Fehlleitung einer Überweisung oder Lastschrift.

Ein wesentliches Ziel in der Datenorganisation muss daher sein, Fehler bei der Arbeit mit Identnummern zu vermeiden bzw. sie zu erkennen, bevor sie zu einer falschen Verarbeitung führen.[6]

1.3 Fehlerarten

Fehler können beim Hören oder Schreiben einer Nummer, beim manuellen oder maschinellen Lesen und Übertragen und beim Eintippen der Nummer geschehen.[7]

Die nachfolgende Tabelle verschafft einen Überblick über die wesentlichen Fehlerarten, die bei der Arbeit mit Nummern vorkommen können.

---

4 Vgl. Grupp, Bruno, Optimale Verschlüsselung bei Online-Datenverarbeitung: Aufbau moderner Nummernsysteme jeder Art, Personennummern und Auftragsnummern, 1. Aufl., Köln 1987, S. 44

5 Vgl. Kunerth, Walter und Werner, Günter, EDV-gerechte Verschlüsselung: Grundlagen und Anwendung moderner Nummernsysteme, 2. Aufl., Stuttgart 1981, S. 28

6 Vgl. Kunerth, Walter, a.a.O., S. 28

7 Vgl. Grupp, Bruno, a.a.O., S. 45

Die häufigsten Fehlerarten bei Nummern sind:[8]

Fehlerart	**Beispiel**	**Häufigkeit**	**Erläuterung**
Eine Ziffer falsch	34567**3** → 34567**8**	Mit 43-77 % aller Fehler die häufigste Fehlerart	Entsteht durch falsche Übertragung oder durch schlecht lesbare Ziffern
Eine oder mehrere Ziffern zu wenig	3**456**73 → 3**46**73	5-37 % aller Fehler	Problem bei variabler Nummernlänge
Eine oder mehrere Ziffern zu viel	3456**73** → 3456**783**	4-10 % aller Fehler	Problem bei variabler Nummernlänge
Einfache Drehfehler	34**56**73 → 34**65**73	3-7 % aller Fehler	Entsteht oft durch die nicht streng an der Ziffernreihenfolge orientierten Aussprache von Zahlen in der deutschen Sprache
Doppelte Drehfehler	**3456**73 → **5634**73	0,2-1,3 % aller Fehler	Hier spielt die Gewohnheit, Ziffernpaare zu sprechen, ebenfalls eine Rolle
Sonstige Drehfehler	3**456**73 → 3**654**73	5-15 % aller Fehler	Vertauschungen über mehrere Stellen hinweg aus verschiedenen Ursachen
Zwei Ziffern falsch	**3**45**6**73 → **8**45**5**73	k. A.	Fehlerart, die oft nicht explizit aufgeführt wird
Gleicher Fehler bei zwei oder mehr aufeinanderfolgenden Stellen	3**456**73 → 3**789**73	k. A.	Spezielle Art von Bündelfehler; oft durch Vertauschung von Zeilen oder Spalten auf Tastaturen
Eine oder mehrere Nullen zu viel oder zu wenig	345**000** → 345**00**	k. A.	Vorwiegend Operativdaten

1.4 Fehlerquellen

Zu den häufigsten Gründen für Fehler bei der manuellen Dateneingabe zählen vor allem außerordentlich lange Nummern, fehlende Gliederungsmittel in langen Nummern, keine Gliederung auf Erfassungsformularen und Bildschirmmasken und eine variable Stellenzahl einer Nummer innerhalb einer maximalen Feldlänge.[9]

---

8 Vgl. Kunerth, Walter, a.a.O., S. 28-29, **und** Jahnke, Bernd, Gestaltung leistungsfähiger Nummernsysteme für die DV-Organisation, 1. Aufl., München 1979, S. 151

9 Vgl. Grupp, Bruno, a.a.O., S. 45

Eine Verdoppelung der Stellenanzahl hat eine Verfünffachung des Fehleranteils zur Folge. Dies liegt daran, dass sich mit steigender Stellenanzahl die Merkbar- und Sprechbarkeit verschlechtert und damit die Gefahr des Verlesens und Verschreibens überproportional steigt.[10]

Verwechslungen resultieren auch aus unkorrekter oder amerikanischer Schreibweise und sonstigen persönlichen Bedingungen (z. B. Erkältung, Müdigkeit) oder Umwelteinflüssen (z. B. Blendung).[11]

Beim maschinellen Lesen und Übertragen von Nummern sind die Fehlerquellen hingegen darin zu finden, dass eine Ziffer falsch erkannt oder ein Bit falsch übertragen wird. Moderne Schriftenlesesysteme erkennen handschriftlich eingetragene Werte. Undeutlich geschriebene Zeichen können hier fehlinterpretiert werden. Bei der Datenübertragung lassen sich Geräte- und Systemstörungen nicht ausschließen.

1.5 Fehlervermeidung durch organisatorische Maßnahmen

Mit organisatorischen Maßnahmen kann die Fehlerquote verringert werden. Um von vornherein die Fehlerquellen klein zu halten, sollte auf möglichst kurze Nummern geachtet werden. Konstante Nummernlängen verhindern, dass eine Zahl zu viel oder zu wenig eingegeben wird, sofern führende Nullen vermieden werden. [12]

Gliederungsmittel wie Bindestriche und Leerstellen unterteilen eine Nummer optisch in Abschnitte und machen sie damit übersichtlicher. Alphanumerische Nummern wirken außerdem fehlervermindernd, wenn bestimmte Gestaltungsregeln eingehalten werden.[13]

Die Ergänzung der Nummer mit zusätzlichen Kontrolleintragungen (z. B. der erste Buchstabe des Nachnamens oder der Preis des Produktes) dient dazu, Fehler bei der Eingabe zu erkennen. Die gleiche Funktion üben Plausibilitätsprüfungen bei Nummerneingaben aus, soweit das verwendete Nummernsystem dies unterstützt. Über sinnvolle Vergleichsprüfungen bei der Online-Datenerfassung können ungültige Nummern abgefangen werden.[14]

Im Zahlungsverkehr der Banken und Sparkassen wird neben der Kontonummer auch der Name des Zahlungsempfängers auf dem Beleg weitergegeben. Es ist möglich, die angegebenen Kontonummern durch Namensvergleich anhand von Listen auf ihre

---

[10] Vgl. Kunerth, Walter, a.a.O., S. 29
[11] Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 485
[12] Vgl. Grupp, Bruno, a.a.O., S. 46
[13] Vgl. Grupp, Bruno, a.a.O., S. 46
[14] Vgl. Grupp, Bruno, a.a.O., S. 46

Richtigkeit zu prüfen.[15] Dies ist nur bei der kontoführenden Sparkasse oder Bank möglich und bedeutet einen erheblichen Arbeitsaufwand. Deshalb finden solche Kontrollen erst ab bestimmten Betragshöhen statt.

Eine weitere gängige Sicherheitsmaßnahme im Zahlungsverkehr ist die Bildung von Kontrollsummen. Bankleitzahlen und Kontonummern werden jeweils addiert, so dass in weiteren Arbeitsgängen eine summenmäßige Abstimmung erfolgen kann. Kompensationsfehler sind hierbei zwar nicht ausgeschlossen, dennoch werden eine Reihe von Fehlerquellen (z. B. fehlende Datensätze) ausgeschaltet.[16]

Sparkassen und Banken bemühen sich, möglichst viele Belege schon vor der Aushändigung an ihre Kunden mit der betreffenden Kontonummer zu versehen. Hiermit lassen sich Schreibfehler von vornherein vermeiden.[17] An den modernen SB-Geräten in den Bankfoyers wird die Kontonummer von der Kundenkarte eingelesen, so dass hier eine Fehleingabe gar nicht erst möglich ist.

Ähnlich verhält es sich mit der Verwendung von Strichcodes, auch Balken- oder Barcodes genannt. Diese werden im Transport, bei Lagerungen und bei Endverbrauchermärkten verwendet, da spezielle Lesegeräte im Vergleich zur manuellen Eingabe schneller und zuverlässiger arbeiten. Durch Verwendung solcher Strichcodes ist es möglich, Daten am Ort ihres Entstehens zu erfassen und Übermittlungsfehler zu vermeiden.[18] In diesem Zusammenhang ist darauf zu achten, dass optische Scanner eine staubarme Umgebung benötigen, da ansonsten weitere Fehler entstehen könnten.

Beim Bildschirmdialog ist es möglich, direkt nach Eingabe einer Identnummer die Personen- oder Objektbezeichnung für eine Sichtprüfung anzuzeigen.[19] Dies setzt voraus, dass die Eingabe von Nummern durch geschultes Personal erfolgt, das anhand der Bildschirmanzeige die Fehler sofort richtig erkennen kann. Außerdem sollten Störfaktoren vermieden werden, z. B. Licht, das sich im Bildschirm spiegelt.

Die genannten Maßnahmen sind geeignet, die Fehlerquote zu verringern. In den meisten Fällen reicht dies nicht aus. Deshalb wurden im Laufe der Zeit unterschiedliche Fehlererkennungsmethoden entwickelt. Nachdem eine Nummer als falsch erkannt wurde muss die richtige Nummer ermittelt werden. Dies bedeutet einen Arbeitsaufwand, der Kosten verursacht. So lange die Kosten eines Fehlers höher sind als die des Korrekturaufwandes, werden Fehlererkennungsmethoden eingesetzt.

---

15 Vgl. o.V., Einheitliches Kontonummernsystem der Sparkassen und Girozentralen (EKONS), Hrsg. Deutscher Sparkassen- und Giroverband e.V. Bonn, 1. Aufl., Stuttgart 1967, S. 21
16 Vgl. o.V., EKONS, a.a.O., S. 21
17 Vgl. o.V., EKONS, a.a.O., S. 21
18 Vgl. Grupp, Bruno, a.a.O., S. 46
19 Vgl. Grupp, Bruno, a.a.O., S. 49

1.6 Fehlererkennungsmethoden

1.6.1 Tabellenkontrolle

Die Tabellenkontrolle kann bei lückenhaften Nummernsystemen Anwendung finden. Lückenhafte Nummernsysteme nutzen aus organisatorischen Gründen nur einen kleinen Teil der theoretisch darstellbaren Nummern aus.

Dies ist vor allem bei hierarchisch gegliederten Nummern der Fall. Zum Beispiel enthält die Internationale Artikelnummer (EAN) Stellen für das Land, den Hersteller und das Produkt. Einige Herstellernummern bzw. Artikelnummern sind nicht belegt.[20]

Die zu prüfende Nummer wird mit den vergebenen Nummern verglichen. Ist keine Identität herzustellen, dann liegt ein Fehler vor.[21]

Je stärker das Nummernsystem ausgenutzt ist, desto geringer ist der Schutz dieses Verfahrens. Außerdem erfordert die Tabellenkontrolle den Zugriff auf die entsprechenden Daten, oder das verwendete Programm muss über eine Kopie dieser Daten verfügen.

Im Supermarkt wird zu der eingelesenen Artikelnummer die Produktbeschreibung herausgesucht und auf dem Kassenbon ausgegeben. Existiert die Nummer jedoch nicht im Kassencomputer, dann ertönt ein Warnsignal.

Ein weiteres Beispiel stammt aus dem Bereich des Homebanking. Die meisten Softwareprodukte verfügen über eine eigene Bankleitzahlentabelle. Bei Eingabe einer gültigen Bankleitzahl wird der Bankname eingesetzt, anderenfalls liegt ein Fehler vor, und die Eingabe wird nicht akzeptiert. Die Bankleitzahlentabelle muss regelmäßig aktualisiert werden.

1.6.2 Kombination von Informationen

Eine weitere Möglichkeit für die Fehlererkennung ist die Kombination von Informationen, die zueinander in einer festen Beziehung stehen, wie Artikelnummer und Preis.[22]

Solche kombinierten Informationen können sehr problematisch sein. Sie führen zu redundanten Daten und bieten oft nur eine geringe Sicherheit, da beispielsweise derselbe Preis bei mehreren Artikeln vorkommen kann. Außerdem bereiten Preisände-

---

[20] Die EAN wird in Kapitel 3.2 genauer beschrieben.
[21] Vgl. Jahnke, Bernd, a.a.O., S. 160
[22] Vgl. Kunerth, Walter, a.a.O., S. 29

rungen in diesem Nummernsystem Schwierigkeiten. Hinzu kommt, dass der Zugriff auf die Stammdaten gewährleistet sein muss.

Eine gute Kombination könnte das Mitführen des Empfängernamens bei einer Banküberweisung sein. Da jedoch Namen keine einheitliche Länge haben und teilweise unterschiedlich geschrieben sein könnten, ist dies nicht praktikabel. Beim Kontonummer-Namens-Vergleich wird auf den Empfängernamen innerhalb des Datensatzes zugegriffen. Dieses Verfahren ist jedoch sehr aufwändig und wird nur in Ausnahmefällen durchgeführt.

Ähnlich verhält es sich beim Mitführen von Namensbestandteilen in der Nummer, etwa des Anfangsbuchstabens.[23] Zum Beispiel gäbe es die Nummern 4711P und 7411M (Kunde Müller), würde der Zahlendreher 4711M als falsch erkannt. Ein Fehler kann jedoch erst zu dem Zeitpunkt festgestellt werden, zu dem ein Zugriff auf die Stammdaten erfolgt. Außerdem ist die Fehlererkennungswahrscheinlichkeit bei häufig vorkommenden Anfangsbuchstaben sehr gering.

1.6.3 Prüfzeichen

Gemäß DIN ist ein Prüfzeichen "im Sinne der Nummerung ein an eine Nummer angehängtes Zeichen zum Prüfen ihrer Richtigkeit."[24]

Die zu schützende Nummer wird um eine Stelle erweitert, die das Prüfzeichen beinhaltet. Es kann sich um ein alphabetisches Zeichen handeln, ist aber bei den meisten verwendeten Verfahren numerisch. Abweichend von der DIN-Norm hat sich im Sprachgebrauch der Begriff *Prüfziffer* durchgesetzt.

Die Prüfziffer wird nach bestimmten Rechenvorschriften aus der ursprünglichen Nummer ermittelt und an die Nummer zur Kontrolle ihrer Richtigkeit angehängt. Damit kann geprüft werden, ob die vorangehende Ziffernfolge nach den Regeln der Rechenvorschrift gebildet ist.[25]

Nicht feststellen lassen sich jedoch Verwechslungen von zwei gültigen Nummern.[26]

---

[23] Vgl. Kunerth, Walter, a.a.O., S. 29

[24] o.V., DIN 6763: Nummerung: Allgemeine Begriffe, Berlin und Köln 1972, Bl. 1 S. 3 **auch** Kunerth, Walter, a.a.O., S. 221

[25] Vgl. Schneider, Wolfgang und Lauterbach, Heinz, Betriebliche Nummernsysteme für die elektronische Datenverarbeitung: Wie verschlüsselt man verarbeitungsgerecht die wesentlichen Betriebsdaten in Wirtschaft und Verwaltung?, 1. Aufl., Heidelberg 1971, S. 40

[26] Vgl. Grupp, Bruno, a.a.O., S. 48

Die Verwendung von Prüfziffern und -buchstaben ist in der heutigen Massendatenverarbeitung weit verbreitet.[27] Man spricht vereinzelt auch von selbstprüfenden oder prüfbaren Nummern.

Der Vorteil eines Prüfziffernverfahrens besteht darin, dass die Nummer jederzeit anhand der Rechenvorschrift überprüft werden kann. Ein Zugriff auf Stammdaten oder Tabellen ist nicht notwendig.

1.6.4 Multiplikatorverfahren

Bei diesem Verfahren wird die gesamte Nummer durch Multiplikation mit einem konstanten Wert und der Addition einer Konstanten verändert. Im Gegensatz zum Prüfziffernverfahren ist die zu sichernde Nummer innerhalb der gesicherten Nummer nicht mehr explizit erkennbar.[28]

Als geeignete Multiplikatoren werden die Zahlen 11, 91, 97, 99, 101, 991, 997, 999 und 1001 genannt.[29] Die zu addierende Konstante wird bei ganzen Zahlen nicht benötigt und kann daher 0 betragen.

Die ursprüngliche Nummer wird ermittelt, indem die übermittelte oder eingegebene Nummer durch den ursprünglichen Multiplikator geteilt wird. Bleibt nach der Division ein Rest, dann liegt ein Fehler vor.

In der Praxis haben sich Multiplikatorverfahren nicht durchgesetzt, weil die ursprüngliche Nummer vor ihrer Weiterverarbeitung immer erst rückverwandelt werden muss.[30]

---

[27] Die einzelnen Prüfziffernverfahren werden im Kapitel 3 genauer behandelt.
[28] Vgl. Jahnke, Bernd, a.a.O., S. 229
[29] Vgl. Jahnke, Bernd, a.a.O., S. 231
[30] Vgl. Jahnke, Bernd, a.a.O., S. 234

2. Anforderungen an ein Prüfziffernverfahren

Ein verwendetes Prüfziffernverfahren soll leicht nachvollziehbar sein, darf keinen unangemessen hohen Aufwand verursachen, soll aber trotzdem nach Möglichkeit alle falschen Nummern erkennen.

Die verschiedenen Prüfziffernverfahren unterscheiden sich hauptsächlich durch die Fehlererkennungswahrscheinlichkeit, die Art und Zahl der Prüfzeichen, die Komplexität des Rechenverfahrens und die Anwendbarkeit auf bereits bestehende Nummernsysteme.[31]

Wie aufwändig ein Prüfziffernverfahren sein darf und ob überhaupt eines zur Anwendung kommt hängt nicht zuletzt von wirtschaftlichen Überlegungen ab. Die Kosten und die Wahrscheinlichkeit eines Fehlers stehen dem Aufwand gegenüber, der durch die längeren Nummern und die zusätzliche Rechenzeit entsteht. Deshalb finden die besonders sicheren mehrstelligen Prüfziffern nur selten Anwendung.

So haben Telefonnummern beispielsweise keine Prüfziffer. Die Verwendung einer falschen Telefonnummer wird schnell erkannt und verursacht nur geringe Kosten in Form einer Telefoneinheit. Die Verwechslung einer Faxnummer könnte dazu führen, dass vertrauliche Informationen in falsche Hände geraten. Die Wahrscheinlichkeit, mithilfe der falschen Nummer ein Faxgerät zu erreichen, ist jedoch gering.

Kfz-Kennzeichen kommen ebenfalls ohne Prüfziffern aus. Die alphanumerische Gliederung wird für ausreichend gehalten. Zur besseren Identifikation wird stets der Autotyp mitnotiert, um Verwechslungen auszuschließen. Ein zusätzliches Zeichen auf dem Nummernschild würde zu einer kleineren Schrift führen, die wiederum schlechter lesbar wäre.

Artikelnummern und Bankkonten kommen dagegen keinesfalls ohne eine Prüfziffer aus, da hier unerkannte Verwechslungen direkt zu finanziellen Schäden führen könnten.

Wegen des hohen Sicherheitsbedürfnisses haben auch Geldscheinseriennummern eine Prüfziffer. Die Seriennummern können also auf ihre Gültigkeit hin überprüft werden. Einzelheiten über die Geldscheinseriennummern werden von der Bundesbank jedoch nicht veröffentlicht.[32]

---

[31] Vgl. Grupp, Bruno, a.a.O., S. 49

[32] Auskunft per E-Mail durch Herrn Wolfgang Hintze, Pressestelle der Deutschen Bundesbank vom 22.07.2002

2.1 Stellenanzahl

Mit einer einzelnen angehängten Prüfziffer kann die Fehlererkennungsrate bei einer zehnstelligen Zahl im günstigsten Fall bereits über 98 % betragen.[33]

Die Restfehlerzahl lässt sich durch die Verwendung mehrerer Prüfstellen minimieren. Die Fehlererkennungswahrscheinlichkeit beträgt bei zweistelligen Prüfziffernverfahren bestenfalls bis zu 99,98 %, bei dreistelligen Verfahren sind nur noch minimale Verbesserungen zu beobachten.[34]

Der Sicherheitsgrad erhöht sich bei mehrstelligen Prüfziffernverfahren nur geringfügig. Es entsteht ein Mehraufwand für die Rechenzeit, die Datenerfassung und -haltung. Der Unterschied in der Fehlererkennungsquote rechtfertigt diesen Mehraufwand im Regelfall nicht.[35] Daher sind in der Praxis überwiegend einstellige Prüfziffernverfahren anzutreffen.

Anders sieht es bei besonders langen Nummern aus, da die Fehlererkennungsrate bei zunehmender Stellenanzahl abnimmt. Eine einstellige numerische Prüfziffer schützt eine neun- bis zehnstellige Grundnummer je nach Verfahren sehr gut. Jede weitere Stelle verschlechtert jedoch dieses Ergebnis.

Da bei besonders langen Nummern eine weitere Prüfstelle in der Datenhaltung nicht so sehr ins Gewicht fällt, finden zweistellige Prüfziffernverfahren Verwendung.

Hierbei kann es sich um zwei einstellige Prüfziffern handeln, die optimal aufeinander abgestimmt ermittelt werden, um so die höchstmögliche Fehlererkennung zu ermöglichen.

Es gibt andererseits Prüfziffernverfahren, die von vornherein eine zweistellige Prüfziffer als Ergebnis liefern und hervorragende Fehlererkennungseigenschaften bieten.

Die in der Literatur behandelten dreistelligen Prüfziffernverfahren haben keine nennenswerte Verbreitung.

2.2 Modulus-Wert

Die Prüfziffernverfahren werden in der Literatur nach dem Modulus klassifiziert, der der jeweiligen Berechnungsmethode zugrunde liegt. Man spricht dann von einem

[33] Vgl. Jahnke, Bernd, a.a.O., S. 164
[34] Vgl. Jahnke, Bernd, a.a.O., S. 208-209
[35] Vgl. Kunerth, Walter, a.a.O., S. 32

Modulus-m-Verfahren.[36] Eine weitere Unterscheidung ist nach den Gewichtungsfolgen und den Modifikationen, etwa einer Quersummenbildung, möglich.

2.2.1 Modulus-11-Verfahren

Die größte Verbreitung haben die Modulus-11-Verfahren. Hier werden alle Stellen der zu prüfenden Nummer mit einem ihnen zugewiesenen Gewichtungsfaktor multipliziert und die Ergebnisse addiert.

Diese Summe wird durch 11 geteilt und der Divisionsrest wiederum von 11 abgezogen. Das Ergebnis ist die anzuhängende Prüfziffer. Da die 11 eine Primzahl ist, verspricht man sich von diesen Verfahren die besten Fehlererkennungseigenschaften.

Verbleibt nach der Division kein Rest, dann ist die Prüfziffer eine 0.

Die Modulus-11-Verfahren führen bei einem Divisionsrest von 1 jedoch zu einer speziellen Darstellungsproblematik, da die errechnete Prüfziffer 10 und damit zweistellig wäre. Hier gibt es verschiedene Lösungsansätze[37]:

1. Alle Nummern, die zu einer Prüfziffer von 10 führen würden, werden nicht vergeben. Dies verringert den verwendbaren Nummernvorrat um ca. 9,1 %.

2. Es wird ein besonderes Zeichen vergeben, z. B. ein "X" bei der ISBN-Nummer. Somit kann der Nummernvorrat voll ausgeschöpft werden.

3. Der Nummer wird eine 1 vorangestellt, sonst eine 0.

4. Statt der 10 wird eine andere einstellige Prüfziffer ein zweites mal vergeben, z. B. die 0. Auch hier kann der Nummernvorrat voll ausgeschöpft werden.

5. Bei einem Divisionsrest von 0 wird keine Prüfziffer vergeben, bei einem Divisionsrest von 1 die dadurch noch verfügbare Ziffer 0.

6. Statt einer einstelligen wird eine zweistellige Prüfziffer angehängt.

Nur die ersten beiden Möglichkeiten erlauben bei der Prüfung einer Nummer mit Prüfziffer folgende Vereinfachung: Die Stellen der Nummer werden wie oben beschrieben gewichtet und summiert. Die angegebene Prüfziffer wird hinzuaddiert. Das Ergebnis muss jetzt ohne Rest durch den Modulus teilbar sein.

---

[36] Vgl. Jahnke, Bernd, a.a.O., S. 171-172
[37] Vgl. Jahnke, Bernd, a.a.O., S. 184

Bei der vierten Variante kann die Prüfziffer 0 sowohl eine *echt*e als auch eine *modifizierte* 0 sein. Dies geht zulasten der Fehlererkennungswahrscheinlichkeit. Die entsprechenden Verfahren werden in der Praxis mit dem Zusatz "modifiziert" gekennzeichnet.

Der dritte, fünfte und sechste Lösungsansatz führen zu unterschiedlichen Nummernlängen. Bei einer in der elektronischen Datenverarbeitung festgelegten Datenfeldlänge sind diese Ansätze deswegen nicht praktikabel.

2.2.2 Modulus-10-Verfahren

Insbesondere bei nachträglicher Einführung eines Prüfziffernverfahrens sind aufgrund der oben beschriebenen Problematik, was bei einem Divisionsrest von 1 zu tun ist, die Modulus-10-Verfahren besser geeignet.[38] Auch bei diesen wird jede Stelle der Nummer mit einem Gewichtungsfaktor multipliziert und die Produkte (oder deren Quersumme) addiert. Diese Summe wird durch 10 geteilt und der Divisionsrest von 10 abgezogen. Das Ergebnis wird als Prüfziffer an die Nummer angehängt. Ist die errechnete Prüfziffer 10, dann wird die Zehnerstelle weggelassen und nur die Ziffer 0 angehängt.

Die Modulus-10-Verfahren erreichen fast genauso gute Fehlererkennungseigenschaften wie die Modulus-11-Verfahren, sofern die Gewichte relativ prim zu 10 sind,[39] also mit dem Modulus-Wert 10 keinen echten gemeinsamen Teiler haben. Der Nummernvorrat kann stets voll ausgeschöpft werden. Eine gleichmäßige Verteilung der Prüfziffern 0 bis 9 ist gegeben.

2.2.3 Modulus-7-Verfahren und Modulus-9-Verfahren

Die Modulus-7-Verfahren werden mit relativ kleinem Modulus auf Dezimalziffern angewendet. Die Berechnung erfolgt ähnlich wie bei den Modulus-10- und Modulus–11-Verfahren. Die Fehlererkennungseigenschaften haben folgende Einschränkungen bei Einzeichenfehlern: Verfälschungen einer 0 in eine 7, einer 1 in eine 8 und einer 2 in eine 9 werden nicht erkannt.[40] Obwohl die 7 eine Primzahl ist, haben diese Verfahren eine schlechte Fehlererkennungswahrscheinlichkeit.

---

38 Vgl. o.V., EKONS, a.a.O., S. 22
39 Vgl. Jahnke, Bernd, a.a.O., S. 179-182
40 Vgl. Jahnke, Bernd, a.a.O., S. 176

Dies wird durch ein **Beispiel** verdeutlicht: Beim einfachen Modulus-7-Verfahren ist die Nummer korrekt, wenn die Nummer mit Prüfziffer ohne Rest durch 7 teilbar ist. Die Prüfung der Nummer 16415 ergibt:

16415 MOD 7 = **0**

Werden die Ziffern 7, 8 und 9 in der Nummer zugelassen, dann wird der folgende Einzeichenfehler nicht erkannt:

86415 MOD 7 = **0**

Die Modulus-9-Verfahren erkennen eine Verfälschung einer 0 in eine 9 nicht[41] und sind daher ebenfalls nur bedingt empfehlenswert.

2.2.4 Modulus-23-Verfahren und Modulus-26-Verfahren

Die in der Praxis wenig beachteten Modulus-23-Verfahren weisen sehr gute Fehlererkennungseigenschaften auf, da die Zahl 23 eine Primzahl ist und ein höherer Modulus-Wert tendenziell eine höhere Sicherheitsrate bietet. Bei Verwendung von Buchstaben reicht eine Prüfstelle aus.[42]

Ein Modulus-26-Verfahren nutzt das ganze Alphabet als Prüfzeichen aus. Die Zahl 26 ist zwar keine Primzahl, bietet aber aufgrund des höheren Modulus-Wertes und des größeren verwendeten Zeichenvorrates eine höhere Sicherheit als das Modulus-23-Verfahren. [43]

In rein numerischen Feldern können diese beiden Verfahren jedoch nicht angewendet werden.

2.2.5 Modulus-97-Verfahren

Die Zahl 97 ist als Modulus-Wert sehr gut geeignet, weil sie eine Primzahl und nur unwesentlich kleiner als 100 ist. Es können zwei Dezimalstellen als Prüfziffern verwendet werden.[44]

---

41 Vgl. Jahnke, Bernd, a.a.O., S. 177
42 Vgl. Jahnke, Bernd, a.a.O., S. 188
43 Vgl. Jahnke, Bernd, a.a.O., S. 191
44 Vgl. Jahnke, Bernd, a.a.O., S. 200

Beim einfachen Modulus-97-Verfahren ist die Nummer mit Prüfziffer insgesamt ohne Rest durch 97 teilbar. Sinnvoll wäre auch die Festlegung, dass der Rest 1 sein muss. Damit kann verhindert werden, dass als Prüfziffer zwei Nullen ermittelt werden.

Bei Verwendung von Gewichtungsfaktoren können diese sowohl einstellig als auch zweistellig sein. In letzterem Fall werden jeweils Ziffernpaare mit den zweistelligen Gewichtungsfaktoren multipliziert. Die Produkte werden addiert und die Summe durch 97 geteilt. Der Rest wird von 97 abgezogen und ergibt die zweistellige Prüfziffer.

2.2.6 Modulus-99-Verfahren

Ein weiteres Prüfziffernverfahren mit zwei Stellen ist das Modulus-99-Verfahren. Die Nummer wird von rechts her in Ziffernpaare aufgeteilt. Diese Ziffernpaare werden addiert. Hat die Summe mehr als zwei Stellen, dann wird dieser Schritt so oft wiederholt, bis eine zweistellige Zahl übrigbleibt. Diese wird als zweistellige Prüfziffer an die Nummer angehängt.[45]

Sollten zwei dieser Ziffernpaare miteinander vertauscht worden sein (Doppeldrehfehler), dann würde dieser Fehler nach der Addition nicht mehr erkannt werden. Somit ist die Fehlererkennungsrate dieses Verfahrens entsprechend geringer als bei dem oben beschriebenen Modulus-97-Verfahren.

2.2.7 Modulus-101-Verfahren

Die Zahl 101 ist eine Primzahl und bietet dadurch hervorragende Fehlererkennungseigenschaften. Beim einfachen Modulus-101-Verfahren wird die Nummer durch 101 geteilt und der Rest von 101 abgezogen. Das Ergebnis ist die zweistellige Prüfziffer. Bei einem Divisionsrest von 0 und 1 gelten die gleichen Lösungsansätze wie beim einstelligen Modulus-11-Verfahren.[46]

2.2.8 Dreistellige Prüfziffernverfahren

Im Folgenden werden drei Verfahren vorgestellt, die eine dreistellige Prüfziffer ermitteln. Durch den sehr hohen Modulus-Wert wird die Restfehlerzahl weiter verringert, jedoch finden diese Verfahren in der Praxis kaum Beachtung.

---

[45] Vgl. Jahnke, Bernd, a.a.O., S. 202
[46] Vgl. Jahnke, Bernd, a.a.O., S. 203

Das Modulus-997-Verfahren wird analog dem Modulus-97-Verfahren gehandhabt. Da die Zahl 997 eine Primzahl ist, bietet dieses Verfahren hervorragende Fehlererkennungseigenschaften.[47]

Beim dreistelligen Modulus-999-Verfahren werden analog dem Modulus-99-Verfahren Dreiergruppen addiert.[48] Wie schon das Modulus-99-Verfahren bietet auch dieses Verfahren keinen guten Schutz gegen Vertauschungen von Ziffern. Eine Vertauschung über drei Stellen hinweg bliebe aufgrund der Addition der dreistelligen Zahlen unerkannt.

Das dreistellige Modulus-1001-Verfahren kann analog zu dem Modulus-11-Verfahren und dem Modulus-101-Verfahren konzipiert werden.[49] Zu beachten ist, dass die Zahl 1001 gegenüber 11 und 101 keine Primzahl ist und bei einem Divisionsrest von 1 die gleichen Probleme wie bei dem Modulus-11-Verfahren bestehen.

2.3 Gewichtung der Ziffern

2.3.1 Modulus-11-Verfahren ohne Gewichtung

Beim einfachen Modulus-11-Verfahren ist die Nummer mit Prüfziffer ohne Rest durch 11 teilbar. Hierbei gilt folgende Kongruenz: "Eine Dezimalzahl ist dann ohne Rest durch 11 teilbar, wenn dies auch für die mit alternierenden Vorzeichen gebildete Quersumme der Zahl gilt."[50]

> Hierzu ein **Beispiel**: Die Nummer 91234 ist ohne Rest durch 11 teilbar. Die Prüfung mit alternierenden Vorzeichen geschieht folgendermaßen:
>
> 9 * (+1) + 1 * (-1) + 2 * (+1) + 3 * (-1) + 4 * (+1) = 11
>
> 11 MOD 11 = **0**

Drehfehler über zwei Stellen hinweg und Doppeldrehfehler werden durch die sich wiederholenden Gewichte nicht erkannt.

---

47 Vgl. Jahnke, Bernd, a.a.O., S. 205
48 Vgl. Jahnke, Bernd, a.a.O., S. 206
49 Vgl. Jahnke, Bernd, a.a.O., S. 206
50 Jahnke, Bernd, a.a.O., S. 174

Dies zeigen zwei weitere **Beispiele**:

Bei der obigen Nummer werden die zweite und vierte Stelle vertauscht. Die falsche Nummer 9**3**2**1**4 wird geprüft:

9 * (+1) + 3 * (-1) + 2 * (+1) + 1 * (-1) + 4 * (+1) = 11

11 MOD 11 = **0**

Bei der obigen Nummer werden die ersten beiden Ziffernpaare vertauscht. Die falsche Nummer **2391**4 wird geprüft:

2 * (+1) + 3 * (-1) + 9 * (+1) + 1 * (-1) + 4 * (+1) = 11

11 MOD 11 = **0**

2.3.2 Modulus-10-Quersummenverfahren

Das einfache Modulus-10-Verfahren ist das Quersummenverfahren. Die Quersumme der Nummer mit Prüfziffer ist ohne Rest durch 10 teilbar. Drehfehler werden bei diesem Verfahren in keinem Fall erkannt.[51]

Dies veranschaulicht folgendes **Beispiel**:

Die Nummer 12340 soll geprüft werden. Die Quersumme der Nummer errechnet sich sehr einfach:

1 + 2 + 3 + 4 + 0 = 10

10 MOD 10 = **0**

Jede andere Ziffernreihenfolge führt zum selben Ergebnis.

Der Vorteil liegt in der Einfachheit. Die Gültigkeit einer Nummer kann jederzeit ohne Hilfsmittel (insbesondere ohne Taschenrechner) festgestellt werden. Das Quersummenverfahren deckt Einzeichenfehler zuverlässig auf und eliminiert damit zumindest die häufigste Fehlerart.

Da jedoch Drehfehler ebenfalls häufig vorkommen können, ist dieses Verfahren für den Einsatz in aktuellen Nummernsystemen nicht ausreichend.

[51] Vgl. Jahnke, Bernd, a.a.O., S. 181

2.3.3 Erkennen von Einzeichenfehlern

Zunächst muss festgestellt werden, dass Einzeichenfehler auch ohne Gewichtungsfaktoren erkannt werden. Um jedoch eine Vielzahl anderer Fehlerarten zu erkennen, werden die Ziffern abhängig von ihrer Stelle gewichtet. Die Gewichte, auch Stellenmultiplikatoren genannt, werden von rechts beginnend auf die Ziffern der Nummer verteilt. Die Gewichte müssen so gewählt sein, dass Einzeichenfehler nach wie vor entdeckt werden können.

Die Gewichte müssen kleiner als der Modulus-Wert sein. Ein Gewicht, das dem Modulus-Wert entspricht, würde die Fehlererkennung unmöglich machen. Auf jeder Nummernstelle sind alle Ziffern zugelassen, die kleiner als der Modulus-Wert sind (i. d. R. 0-9).[52] Werden beispielsweise bei einem Modulus-7-Verfahren die Ziffern 7, 8 und 9 verwendet, dann geht dies zulasten der Fehlererkennungswahrscheinlichkeit, wie weiter oben bereits dargestellt wurde.

Die Gewichte und der Modulus sollten relativ prim zueinander sein, also keinen gemeinsamen Teiler haben. Ferner sollte der Modulus größer als die Stellenanzahl gewählt werden. Damit kann gewährleistet werden, dass Einzeichenfehler sicher erkannt werden.[53]

Das Modulus-11-Verfahren hat den Vorteil, dass alle Zahlen bis 10 als Gewichte verwendet werden können, wie aus folgender Tabelle entnommen werden kann. Die Ergebnisse verdeutlichen, wie sich die gewichteten Ziffern nach Division durch 11 auswirken:

	Ziffern der Nummer									
Gewichte	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**
1	1	2	3	4	5	6	7	8	9	0
2	2	4	6	8	10	1	3	5	7	0
3	3	6	9	1	4	7	10	2	5	0
4	4	8	1	5	9	2	6	10	3	0
5	5	10	4	9	3	8	2	7	1	0
6	6	1	7	2	8	3	9	4	10	0
7	7	3	10	6	2	9	5	1	8	0
8	8	5	2	10	7	4	1	9	6	0
9	9	7	5	3	1	10	8	6	4	0
10	10	9	8	7	6	5	4	3	2	0

[52] Vgl. Jahnke, Bernd, a.a.O., S. 165
[53] Vgl. Jahnke, Bernd, a.a.O., S. 167

Das häufig benutzte Modulus-10-Verfahren kann nur einen Teil der Zahlen als Gewichte sinnvoll verwenden. Die folgende Tabelle stellt dar, wie sich die gewichteten Ziffern auswirken:

	Ziffern der Nummer									
Gewichte	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**
1	1	2	3	4	5	6	7	8	9	0
2	2	4	6	8	0	2	4	6	8	0
3	3	6	9	2	5	8	1	4	7	0
4	4	8	2	6	0	4	8	2	6	0
5	5	0	5	0	5	0	5	0	5	0
6	6	2	8	4	0	6	2	8	4	0
7	7	4	1	8	5	2	9	6	3	0
8	8	6	4	2	0	8	6	4	2	0
9	9	8	7	6	5	4	3	2	1	0

Die Gewichte 2, 4, 5, 6 und 8 sind nicht relativ prim zu 10 und schützen daher nicht ausreichend vor Einzeichenfehlern. Im folgenden Beispiel wird ein Modulus-10-Verfahren mit den Gewichten 2 und 5 angenommen.

Beispiel: Die Nummer ohne Prüfziffer soll 1234 sein. Die Prüfziffer errechnet sich folgendermaßen:

1 * 5 + 2 * 2 + 3 * 5 + 4 * 2 = 32

32 MOD 10 = 2

10 – 2 = **8**

Die korrekte Nummer mit Prüfziffer lautet also: 1234**8**.

Im Folgenden werden die richtige Nummer 12348 und die zwei fehlerhaften Nummern **3**2348 und 1**7**348 geprüft:

Zu prüfende Nummer:	12348	**3**2348	1**7**348
Gewichtung der Ziffern	1 * 5 + 2 * 2 + 3 * 5 + 4 * 2 + 8 = 40	**3** * 5 + 2 * 2 + 3 * 5 + 4 * 2 + 8 = 50	1 * 5 + **7** * 2 + 3 * 5 + 4 * 2 + 8 = 50
Divisionsrest	40 MOD 10 = **0**	50 MOD 10 = **0**	50 MOD 10 = **0**
Ergebnis	Richtig	Richtig	Richtig

Fazit: Die Fehler werden nicht erkannt und die Nummern als gültig interpretiert.

2.3.4 Erkennen von Vertauschungen

Zum sicheren Erkennen der Vertauschung zweier beliebiger Ziffern sollten die Gewichte unterschiedlich gewählt werden. Die Differenz zweier Gewichte sollte relativ prim zum Modulus sein.[54]

Sollte etwa bei einem Modulus-10-Verfahren die Differenz zweier Gewichte 2 betragen und zwei benachbarte Stellen um genau 5 abweichen, dann würden Drehfehler nicht erkannt.

Diese Problematik kann mit einem **Beispiel** verdeutlicht werden:

Bei einem **Modulus-10-Verfahren** sollen die **Gewichte 1 und 3** verwendet werden. Die korrekte Nummer ohne Prüfziffer ist 1678.

Die Prüfziffer wird ermittelt:

1 * 3 + 6 * 1 + 7 * 3 + 8 * 1 = 38

38 MOD 10 = 8

10 – 8 = **2**

Die korrekte Nummer lautet also 1678**2**.

Durch einen Drehfehler entsteht die fehlerhafte Nummer **61**782. Die Prüfung ergibt Folgendes:

6 * 3 + 1 * 1 + 7 * 3 + 8 * 1 + 2 = 50

50 MOD 10 = **0**

Die Summe weicht durch diesen Drehfehler um genau 10 ab. Nach Ermittlung des Divisionsrestes ist der Fehler nicht mehr zu erkennen.

2.3.5 Erkennen von Bündelfehlern

Zum Erkennen der Fehlerart "Gleicher Fehler bei mehreren aufeinander folgenden Stellen" sollten die Größe und die Reihenfolge der Gewichte so gewählt werden, dass die Summe mehrerer benachbarter Gewichte relativ prim zum Modulus ist.[55]

[54] Vgl. Jahnke, Bernd, a.a.O., S. 168
[55] Vgl. Jahnke, Bernd, a.a.O., S. 169

Hierzu ein **Beispiel**.

Angenommen sei ein **Modulus-10-Verfahren** mit den **Gewichten 3, 7, 1**. Die richtige Nummer ohne Prüfziffer soll 12356 sein.

Die Prüfziffer errechnet sich folgendermaßen:

1 * 7 + 2 * 3 + 3 * 1 + 5 * 7 + 6 * 3 = 69

69 MOD 10 = 9

10 – 9 = **1**

Die korrekte Nummer mit Prüfziffer wäre also 12356**1**.

Die mit der Datenerfassung beauftragte Person "verrutscht" jedoch in der Reihe und gibt stattdessen 123**89**1 ein. Die Nummer wird geprüft:

1 * 7 + 2 * 3 + 3 * 1 + 8 * 7 + 9 * 3 + 1 = 100

100 MOD 10 = **0**

Die in diesem Beispiel verwendete Gewichtungsfolge hätte den Nachteil, dass beim gleichen Fehler in zwei benachbarten Stellen die gleiche Prüfziffer errechnet und damit der Fehler nicht erkannt würde. Das ist möglich, weil die Summe aus 3 und 7 dem Modulus-Wert 10 entspricht.

Die Gewichtungsfolge 3, 9, 7, 1 wäre besser geeignet, da keine Addition aus benachbarten Gewichten 10 ergeben würde. Außerdem fänden dann alle Gewichte Verwendung, die beim Modulus-10-Verfahren eingesetzt werden können, um auch Einzeichenfehler noch sicher erkennen zu können..

2.4 Laufende Quersummenbildung

Das Modulus-10-Verfahren mit den Gewichten 1 und 2 erkennt nicht alle Einzeichenfehler auf den mit 2 gewichteten Stellen. Die laufende Quersummenbildung bietet eine Möglichkeit, die Fehlererkennungswahrscheinlichkeit auf praktisch 100 % anzuheben.[56]

Die Stellen der zu prüfenden Nummer werden mit dem ihnen zugewiesenen Gewichtungsfaktor multipliziert. Kommt hierbei ein mehrstelliger Wert heraus, wird die Quersumme ermittelt. Ist das Ergebnis immer noch mehrstellig, so wird hieraus wiederum die Quersumme ermittelt. Die Summe wird also nur aus einstelligen Ergeb-

[56] Vgl. Jahnke, Bernd, a.a.O., S. 183

nissen gebildet. Wie auch bei den anderen Verfahren wird der Divisionsrest vom Modulus subtrahiert und ergibt die Prüfziffer.

Weiter oben wurde festgestellt, dass bei einem **Modulus-10-Verfahren** die **Gewichte 2 und 5** ungeeignet sind, da diese Gewichte nicht prim zu 10 sind. Die laufende Quersummenbildung gleicht diese Schwäche aus, wie dieses **Beispiel** zeigt:

> Die Nummer ohne Prüfziffer soll wie beim obigen Beispiel 1234 sein. Die Prüfziffer errechnet sich folgendermaßen:
>
> 1 * 5 + 2 * 2 + 3 * 5 + 4 * 2
>
> 5 + 4 + (1+5) + 8 = 23
>
> 23 MOD 10 = 3
>
> 10 – 3 = **7**
>
> Die korrekte Nummer mit Prüfziffer lautet also: 1234**7**.

Die in Abschnitt 2.3.3 ermittelte Prüfziffer war 8. Nachdem die laufende Quersummenbildung bereits zu einer anderen Prüfziffer führt, soll nun verglichen werden, ob die Einzeichenfehler zuverlässig erkannt werden.

Im folgenden Beispiel werden die korrekte Nummer und zwei verfälschte Nummern auf ihre Richtigkeit geprüft. Bei den falschen Nummern wurden eine mit 5 gewichtete Stelle um 2 bzw. eine mit 2 gewichtete Stelle um 5 verändert.

Zu prüfende Nummer:	12347	**3**2347	1**7**347
Gewichtung der Ziffern	1 * 5 + 2 * 2 + 3 * 5 + 4 * 2 + 7	**3** * 5 + 2 * 2 + 3 * 5 + 4 * 2 + 7	1 * 5 + **7** * 2 + 3 * 5 + 4 * 2 + 7
Quersummenbildung	5 + 4 + (1+5) + 8 + 7 = 30	**(1+5)** + 4 + (1+5) + 8 + 7 = 31	5 + **(1+4)** + (1+5) + 8 + 7 = 31
Divisionsrest	30 MOD 10 = **0**	31 MOD 10 = **1**	31 MOD 10 = **1**
Ergebnis	Richtig	Falsch	Falsch

Fazit: Der Divisionsrest bei den fehlerhaften Nummern ist ungleich 0, sie werden als falsch erkannt.

Die Auswirkung der Quersummenbildung soll nun näher untersucht werden. In der folgenden Tabelle ist zu sehen, wie sich die gewichteten Ziffern bei laufender Quersummenbildung im Modulus-10-Verfahren auswirken:

	Ziffern der Nummer									
Gewichte	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**
1	1	2	3	4	5	6	7	8	9	0
2	2	4	6	8	1	3	5	7	9	0
3	3	6	9	3	6	9	3	6	9	0
4	4	8	3	7	2	6	1	5	9	0
5	5	1	6	2	7	3	8	4	9	0
6	6	3	9	6	3	9	6	3	9	0
7	7	5	3	1	8	6	4	2	9	0
8	8	7	6	5	4	3	2	1	9	0
9	9	9	9	9	9	9	9	9	9	0

Ohne Quersummenbildung sind nur die Gewichte 1, 3, 7 und 9 geeignet, alle Einzeichenfehler zu erkennen.

Mithilfe der laufenden Quersummenbildung können hingegen die Gewichte 1, 2, 4, 5, 7 und 8 verwendet werden. Die Gewichte 3, 6 und 9 liefern jedoch eine schlechte bis gar keine Unterscheidung in der Berechnung.

Drehfehler mit den Ziffern 9 und 0 werden nicht erkannt, da sie auf jeder Stelle dasselbe Ergebnis erzeugen. Somit ist dieses Verfahren nur dann geeignet, wenn diese Ziffern im Nummernsystem nicht häufig sind. Außerdem werden Drehfehler zwischen den Ziffern 3 und 6 nicht sicher erkannt.

3 Arten von Prüfziffernverfahren

3.1 Die ISBN

Die *Internationale Standard-Buchnummer (ISBN)* ist zehnstellig. An erster Stelle steht die Gruppennummer (z. B. 3 für deutschsprachige Bücher). Die Stellen 2 bis 9 enthalten die Verlagsnummer und die Titelnummer. An zehnter Stelle steht die Prüfziffer.[57]

Die Prüfziffer der ISBN wird nach dem **Modulus-11-Verfahren** ermittelt. Die Ziffern der Nummer ohne Prüfziffer werden von rechts nach links mit den **Gewichtungsfaktoren 2 bis 10** multipliziert und die Ergebnisse addiert. Die Summe wird durch 11 geteilt und der Divisionsrest wiederum von 11 abgezogen. Das Ergebnis ist die Prüfziffer.[58]

Ist das Ergebnis 11, so ist die Prüfziffer 0. Ist das Ergebnis 10, dann wird der Buchstabe X angehängt, die römische Ziffer für 10.

Beispiel: Das Buch "Einführung in die Wirtschaftsinformatik" von Stahlknecht, Hasenkamp hat die ISBN 3-540-41986-1.

Die Berechnung der Prüfziffer erfolgt so:

3 * 10 + 5 * 9 + 4 * 8 + 0 * 7 + 4 * 6 + 1 * 5 + 9 * 4 + 8 * 3 + 6 * 2 = 208

208 MOD 11 = 10

11 – 10 = **1**

Die Prüfung der Nummer sieht dagegen so aus:

3 * 10 + 5 * 9 + 4 * 8 + 0 * 7 + 4 * 6 + 1 * 5 + 9 * 4 + 8 * 3 + 6 * 2 + 1=209

209 MOD 11 = **0**

Da 11 eine Primzahl ist, weist dieses Verfahren hervorragende Fehlererkennungseigenschaften auf.

Keines der verwendeten Gewichte hat einen gemeinsamen Teiler mit 11. Dadurch werden alle Einzeichenfehler sicher erkannt.

Drehfehler, auch über mehrere Stellen hinweg, werden augenscheinlich, weil sich alle Gewichtungsfaktoren voneinander unterscheiden.

---

57 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 148
58 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 486

Lediglich die benachbarten Gewichte 5 und 6 auf den Stellen 5 und 6 beinhalten das Risiko, dass ein gleichartiger Fehler in diesen beiden benachbarten Stellen unerkannt bleibt.[59] Die Summe dieser beiden Gewichte ist 11 und entspricht damit dem Modulus-Wert. Werden diese beiden Stellen beispielsweise um 1 erhöht, so erhöht sich die Summe um 11 und die falsche Nummer wird als gültig erkannt.

Beispiel: Die Prüfung der falschen Nummer 3-540-**52**986-1 sieht so aus:

3 * 10 + 5 * 9 + 4 * 8 + 0 * 7 + **5** * 6 + **2** * 5 + 9 * 4 + 8 * 3 + 6 * 2 + 1=**220**

220 MOD 11 = **0**

Eine Verbesserung könnte erzielt werden, indem die Gewichtungsfaktoren nicht der Reihe nach, sondern unregelmäßig verteilt würden.

3.2 Die EAN

Die *Internationale Artikelnummer*, auch *Europäische Artikelnummer (EAN)* genannt, wurde 1977 für Handelswaren eingeführt.[60]

Die 13stellige Nummer setzt sich aus dem Länderkennzeichen (z. B. 40 bis 44 für Deutschland), der Herstellernummer und der Artikelnummer zusammen. Bei Büchern beginnt die EAN mit 978, gefolgt von der ISBN (ohne Prüfziffer). Die dreizehnte Stelle beinhaltet die Prüfziffer.[61]

Bei Einführung der EAN war es notwendig gewesen, die Rechenmethode des amerikanischen UPC-Systems (Universal Product Code) zu übernehmen, damit beide Systeme kompatibel bleiben.[62] Die UPC unterscheidet sich von der EAN dadurch, dass sie nur 12stellig ist.

Die Prüfziffer der EAN wird nach dem **Modulus-10-Verfahren** ermittelt. Die Ziffern der Nummer ohne Prüfziffer werden von rechts nach links mit den **Gewichtungsfaktoren 3 und 1** multipliziert und die Ergebnisse addiert. Die Summe wird durch 10 geteilt und der Divisionsrest wiederum von 10 abgezogen. Das Ergebnis ist die Prüfziffer.[63]

Ist das Ergebnis 10, so ist die Prüfziffer eine 0.

---

59 Vgl. Jahnke, Bernd, a.a.O., S. 185
60 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 42
61 Vgl. Stahlknecht, Peter und Hasenkamp, Ulrich, a.a.O., S. 149
62 Vgl. Schiebel, Walter, Die Europäische Artikelnummer (EAN): Ausbreitung und Anwendung in Warenwirtschaftssystemen (mit den offiziellen EAN-Richtlinien), 1. Aufl., Wien 1987, S. 224
63 Vgl. Schiebel, Walter, a.a.O., S. 224

Beispiel: Wie bereits oben erwähnt, hat das Buch "Einführung in die Wirtschaftsinformatik" von Stahlknecht, Hasenkamp die ISBN 3-540-41986-1. Folglich lautet die EAN: 97 83540 41986 0

Die Berechnung der Prüfziffer erfolgt:

9 * 1 + 7 * 3 + 8 * 1 + 3 * 3 + 5 * 1 + 4 * 3 + 0 * 1 + 4 * 3 + 1 * 1 + 9 * 3 + 8 * 1 + 6 * 3 = 130

130 MOD 10 = 0

10 – 0 = **10**

Die Prüfung der Nummer sieht dementsprechend so aus:

9 * 1 + 7 * 3 + 8 * 1 + 3 * 3 + 5 * 1 + 4 * 3 + 0 * 1 + 4 * 3 + 1 * 1 + 9 * 3 + 8 * 1 + 6 * 3 + 0 = 130

130 MOD 10 = **0**

In den EAN-Richtlinien ist weiterhin eine achtstellige Kurzversion vorgesehen. Die Prüfziffer wird hier analog ermittelt.[64]

Die Gewichte 1 und 3 sind geeignet, alle Einzeichenfehler zu erkennen, da sie keinen gemeinsamen Teiler mit 10 haben.

Die Summe mehrerer benachbarter Gewichte ergibt in keinem Fall 10, wodurch ein gleicher Fehler auf mehreren benachbarten Stellen in jedem Fall erkannt wird.

Die Differenz der Gewichte ist 2 und damit nicht prim zum Modulus-Wert. Einfache Drehfehler werden nur erkannt, solange zwei benachbarte Ziffern nicht um genau 5 abweichen.

Doppeldreh- und Drehfehler über eine gerade Anzahl an Stellen hinweg werden durch die sich wiederholenden Gewichte dagegen nicht erkannt.

Beispiel: Die Prüfung der falschen EAN: 97 83540 **91**4**8**6 0 zeigt:

9 * 1 + 7 * 3 + 8 * 1 + 3 * 3 + 5 * 1 + 4 * 3 + 0 * 1 + **9** * 3 + 1 * 1 + **4** * 3 + 8 * 1 + 6 * 3 + 0 = 130

130 MOD 10 = **0**

Eine Verbesserung könnte erzielt werden, indem mehr als nur zwei verschiedene Gewichtungsfaktoren Anwendung finden, z. B. 3, 9, 7, 1.

---

[64] Vgl. Schiebel, Walter, a.a.O., S. 224

In Anbetracht der Länge der EAN sollte auch überlegt werden, ob ein zweistelliges Prüfziffernverfahren infrage kommen könnte. Dies würde bedeuten, dass die gesamte Nummer um eine weitere Stelle anwächst. Da die EAN jedoch fast ausschließlich mit Strichcodelesern erfasst wird, wäre der Mehraufwand gering.

Andererseits ist die EAN ein sehr lückenhaftes Nummernsystem, d. h. nicht alle theoretisch möglichen Nummern werden auch verwendet. Dies erlaubt wiederum eine Tabellenkontrolle. Wird zu der eingelesenen Nummer kein Datensatz gefunden, so liegt selbst bei gültiger Prüfziffer ein Fehler vor.

3.3 Prüfziffern bei Bankkontonummern

3.3.1 Modulus 10 mit Gewichtung 2, 1

Der Deutsche Sparkassen- und Giroverband e. V. Bonn (DSGV) schlägt in seinem Merkblatt *"Einheitliches Kontonummernsystem der Sparkassen und Girozentralen (EKONS)"* von Mai 1967 bevorzugt die Verwendung eines Modulus-10-Verfahrens vor.[65]

Nach diesem Merkblatt enthält die **maximal zehnstellige Kontonummer** neben der laufenden Kontonummer auch Stellen für die Kontenklassifizierung und die Betriebsstelle.[66] Die erste Stelle von rechts beinhaltet die Prüfziffer.

Die einzelnen Stellen der Kontonummer ohne Prüfziffer werden von rechts nach links mit den Gewichten 2, 1, 2, 1 usw. multipliziert. Die Produkte werden addiert. Die Summe wird durch 10 geteilt und der Rest von 10 subtrahiert. Das Ergebnis ist die Prüfziffer.[67] Ist die Prüfziffer 10, so wird nur die Ziffer 0 als Prüfziffer verwendet.

Beispiel: Eine Kontonummer ohne Prüfziffer lautet 456789. Die Prüfziffer wird ermittelt:

4 * 1 + 5 * 2 + 6 * 1 + 7 * 2 + 8 * 1 + 9 * 2 = 60

60 MOD 10 = 0

10 – 0 = **10 → 0**

Die vollständige Kontonummer lautet 4567890

[65] Vgl. o.V., EKONS, a.a.O., S. 20
[66] Vgl. o.V., EKONS, a.a.O., S. 12-19
[67] Vgl. o.V., EKONS, a.a.O., S. 23

Dieses Verfahren hat die auffällige Schwäche, dass der Gewichtungsfaktor 2 ein Teiler von 10 ist. Veränderungen um genau 5 werden auf den entsprechenden Stellen nicht erkannt, wie bereits in Kapitel 2.3.3 festgestellt wurde. Deshalb kommt diese Variante in der Praxis nicht vor.

Doppeldrehfehler und Drehfehler über eine gerade Stellenanzahl hinweg werden aufgrund der sich wiederholender Gewichtungsfaktoren nicht erkannt.

3.3.2 Modulus 10 mit laufender Quersummenbildung

Das zweite im oben genannten Merkblatt beschriebene Verfahren ist eine Variante des Modulus-10-Verfahrens mit der Gewichtung 2, 1. Dadurch, dass aus den jeweils zweistelligen Produkten eine Quersumme gebildet wird, wird ein höherer Sicherheitsfaktor erzielt.[68] Diese Variante wird unter der Kennziffer 00 in den *"Prüfzifferberechnungsmethoden zur Prüfung von Kontonummern auf ihre Richtigkeit"* von der Deutschen Bundesbank beschrieben.[69] Sie findet heute Anwendung bei der Mehrzahl der Sparkassen und wenigen kleinen Bankhäusern.

Beispiel: Die Kontonummer ohne Prüfziffer lautet wie oben 456789. Die Prüfziffer wird ermittelt:

4 * 1 + 5 * 2 + 6 * 1 + 7 * 2 + 8 * 1 + 9 * 2

4 + (1+0) + 6 + (1+4) + 8 + (1+8) = 33

33 MOD 10 = 3

10 – 3 = **7**

Die vollständige Kontonummer lautet nun 4567897

Durch die Quersummenbildung erkennt dieses Verfahren alle Einzeichenfehler. Drehfehler zwischen Ziffern 3 und 6 und zwischen 0 und 9 werden nicht erkannt. Die variable Kontonummernlänge begünstigt zudem das Auftreten von Doppeldrehfehlern, die mit dieser Gewichtungsfolge nicht aufgedeckt werden.[70] Drehfehler über eine gerade Anzahl an Stellen hinweg werden aufgrund der sich wiederholenden Gewichte nicht erkannt. Außerdem erlaubt die variable Kontonummernlänge den Fehler, dass zwei Nullen zu viel angehängt oder eingefügt werden.

---

68 Vgl. o.V., EKONS, a.a.O., S. 25

69 Vgl. o.V., Prüfzifferberechnungsmethoden zur Prüfung von Kontonummern auf ihre Richtigkeit, Hrsg. Deutsche Bundesbank Z 11-5, Stand: Juni 2002, http://www.bundesbank.de/zv/zv_pruefziffern.php

70 Vgl. o.V., EKONS, a.a.O., S. 25

Beispiel: An die oben ermittelte Kontonummer werden zwei Nullen angehängt. Die Nummer 456789**700** wird geprüft:

4 * 1 + 5 * 2 + 6 * 1 + 7 * 2 + 8 * 1 + 9 * 2 + 7 * 1 + 0 * 2 + 0

4 + (1+0) + 6 + (1+4) + 8 + (1+8) + 7 + 0 + 0 = 4**0**

Da die Einerzahl des Ergebnisses eine 0 ist, wird die Kontonummer als richtig erkannt.

3.3.3 Modulus 10 mit Gewichtung 3, 7, 1

Dieses Verfahren ist nicht in dem vom Deutschen Sparkassen- und Giroverband herausgegebenen Merkblatt beschrieben, ist jedoch vor allem bei den württembergischen Kreissparkassen in Anwendung.[71]

Die Stellen der Kontonummer werden von rechts nach links mit den Ziffern 3, 7, 1, 3, 7, 1 usw. multipliziert und die Produkte addiert. Die Summe wird durch 10 geteilt und der Rest von 10 subtrahiert. Das Ergebnis ist die Prüfziffer. Bei einem Divisionsrest von 10 ist die Prüfziffer 0.[72]

Beispiel: Eine Kontonummer ohne Prüfziffer lautet 456789. Die Prüfziffer wird ermittelt:

4 * 1 + 5 * 7 + 6 * 3 + 7 * 1 + 8 * 7 + 9 * 3 = 147

147 MOD 10 = 7

10 – 7 = **3**

Die vollständige Kontonummer lautet 456789**3**

Da die hier verwendeten Gewichtungsfaktoren keine Teiler von 10 sind, werden Einzeichenfehler auch ohne die laufende Quersummenbildung sicher erkannt.

Einfache Dreh- und Doppeldrehfehler werden ebenfalls aufgespürt. Sollten jedoch ganze Dreierblöcke vertauscht werden oder ein Drehfehler über drei Stellen hinweg geschehen, dann würde dieser Fehler nicht gefunden werden.

---

[71] Vgl. o.V., Bankleitzahlen-Datei, Hrsg. Deutsche Bundesbank, Stand: Juni 2002, http://www.bundesbank.de/zv/zv_bankleitzahlen.htm

[72] Vgl. o.V., Prüfziffernberechnungsmethoden..., a.a.O., S. 1

Da die Summe der beiden Gewichte 3 und 7 genau dem Modulus-Wert entspricht, besteht die Gefahr, dass ein gleichartiger Fehler bei diesen entsprechenden zwei benachbarten Stellen nicht erkannt wird.

Beispiel: Bei der Eingabe "verrutscht" ein Mitarbeiter bei der zweiten und dritten Stelle auf der Tastatur. Die falsche Nummer 4**23**7893 wird geprüft:

4 * 1 + 2 * 7 + 3 * 3 + 7 * 1 + 8 * 7 + 9 * 3 + 3 = 120

120 MOD 10 = **0**

Außerdem beträgt die Differenz der Gewichte 1 und 3 genau 2. Da 2 ein Teiler von 10 ist, werden Drehfehler zwischen Ziffern, die um 5 differieren, nicht erkannt.

3.3.4 Modulus 11 mit Gewichtung 2, 3, 4, 5, 6, 7, 8, 9, 10

Das Merkblatt des DSGV beschreibt neben dem favorisierten Modulus-10-Verfahren mit laufender Quersummenbildung auch das Modulus-11-Verfahren mit den Gewichten 2 bis 10. Kontonummern, bei denen als Rest 1 entsteht, ergäben eine Prüfziffer von 10 und können daher nicht vergeben werden.[73] Ist der Rest 0, dann ist auch die Prüfziffer 0.

In der Aufstellung der Prüfzifferberechnungsmethoden der Deutschen Bundesbank wird dieses Verfahren unter der Kennziffer 07 beschrieben und wird nur von sehr wenigen Sparkassen eingesetzt.[74]

Beispiel: Eine Kontonummer ohne Prüfziffer lautet 345679. Die Prüfziffer wird ermittelt:

3 * 7 + 4 * 6 + 5 * 5 + 6 * 4 + 7 * 3 + 9 * 2 = 133

133 MOD 11 = 1

11 – 1 = **10**

Diese Kontonummer kann nicht vergeben werden!

Dieses Verfahren bietet sehr gute Fehlererkennungseigenschaften, erlaubt jedoch nicht die lückenlose Vergabe von Kontonummern. Da es rechnerisch weitgehend identisch mit der ISBN-Nummer ist, gilt auch hier die Kritik, dass die Gewichtungsfaktoren 5 und 6 nicht nebeneinander stehen sollten.

---

[73] Vgl. o.V., EKONS, a.a.O., S. 26
[74] Vgl. o.V., Prüfzifferberechnungsmethoden..., a.a.O., S. 2

3.3.5 Modulus 11 mit Gewichtung 2, 3, 4, ... 10 (modifiziert)

Bei Einbeziehung der Hauptbuchkonten der Banken in die Prüfziffernberechnung kann ein Verfahren, das nicht alle Nummern zulässt, nicht verwendet werden. Für Hauptbuchkontonummern gibt es feste Vorgaben, so dass das Modulus-11-Verfahren nur in modifizierter Art anwendbar ist.[75]

Dieses Verfahren sieht deshalb vor, bei einem Divisionsrest von 1 als Prüfziffer die Ziffer 0 ein weiteres Mal zu vergeben, was jedoch zulasten der Fehlererkennung geht.[76]

Beispiel: Wie oben. Die Kontonummer ohne Prüfziffer lautet 345679. Die Prüfziffer wird ermittelt:

3 * 7 + 4 * 6 + 5 * 5 + 6 * 4 + 7 * 3 + 9 * 2 = 133

133 MOD 11 = 1

11 – 1 = **10 → 0**

Als Prüfziffer wird die 0 vergeben. Die Kontonummer lautet nun 3456790.

In der Aufstellung der Prüfzifferberechnungsmethoden der Deutschen Bundesbank wird diese Berechnungsmethode unter der Kennziffer 10 aufgeführt und wird vor allem bei Volksbanken und Raiffeisenbanken eingesetzt.[77]

Das modifizierte Modulus-11-Verfahren erkennt nicht alle Einzeichenfehler, da die Prüfziffer sowohl eine "echte" als auch eine "modifizierte" 0 enthalten kann.[78]

Der Vorteil ist jedoch, dass die Kontonummern lückenlos vergeben werden können. Wie oben gilt auch hier die Kritik, dass die Gewichtungsfaktoren 5 und 6 nicht nebeneinander stehen sollten.

3.3.6 Modulus 11 mit Gewichtung 2, 3, 4, 5, 6, 7 (modifiziert)

Dieses Verfahren findet hauptsächlich bei Volks- und Raiffeisenbanken Anwendung, aber auch z. B. bei der Sparkasse Heidelberg. In der Aufstellung der Prüfzifferberechnungsmethoden der Deutschen Bundesbank wird diese Berechnungsmethode unter der Kennziffer 06 beschrieben.

---

[75] Vgl. o.V., EKONS, a.a.O., S. 27
[76] Vgl. o.V., EKONS, a.a.O., S. 28
[77] Vgl. o.V., Prüfzifferberechnungsmethoden..., a.a.O., S. 3
[78] Vgl. o.V., EKONS, a.a.O., S. 28

Die Prüfziffer wird ermittelt, indem die Stellen der Kontonummer von rechts nach links mit den Gewichten 2, 3, 4, 5, 6, 7 multipliziert und die Produkte addiert werden. Die Summe wird durch 11 geteilt und der Rest von 11 subtrahiert. Das Ergebnis ist die Prüfziffer. Bei einem Ergebnis von 10 oder 11 ist die Prüfziffer 0. [79]

Beispiel: Eine Kontonummer ohne Prüfziffer lautet 123456789. Die Prüfziffer wird ermittelt:

1 * 4 + 2 * 3 + 3 * 2 + 4 * 7 + 5 * 6 + 6 * 5 + 7 * 4 + 8 * 3 + 9 * 2 = 174

174 MOD 11 = 9

11 – 9 = **2**

Die vollständige Kontonummer lautet 1234567892

Das modifizierte Modulus-11-Verfahren erkennt nicht alle Einzeichenfehler, da die Prüfziffer sowohl eine "echte" als auch eine "modifizierte" 0 enthalten kann.[80]

Der Vorteil ist jedoch, dass die Kontonummern lückenlos vergeben werden können. Die Gewichtungsfaktoren 5 und 6 sollten nicht nebeneinander stehen, da gleichartige Fehler auf diesen Stellen sonst nicht erkannt werden.

Hinzu kommt, dass eine Vertauschung der ersten drei Stellen mit den Stellen 7 bis 9 durch die sich wiederholenden Gewichte unerkannt bleibt.

Beispiel: Die falsche Kontonummer lautet **789**456**123**2. Die Nummer wird geprüft:

7 * 4 + 8 * 3 + 9 * 2 + 4 * 7 + 5 * 6 + 6 * 5 + 1 * 4 + 2 * 3 + 3 * 2 + 2 = 176

176 MOD 11 = **0**

Solange Girokontonummern ohne Prüfziffern nur sechs Stellen haben, tritt dieser Schwachpunkt jedoch praktisch nicht auf.

3.4 Die IBAN

Die Internationale Kontonummer (*International Bank Account Number, IBAN*) soll den grenzüberschreitenden Zahlungsverkehr vereinheitlichen.[81] Für Auslandszahlungen ist die IBAN seit 1.1.2002 zwingend vorgeschrieben.

---

79 Vgl. o.V., Prüfzifferberechnungsmethoden..., a.a.O., S. 2
80 Vgl. o.V., EKONS, a.a.O., S. 28

3.4.1 Aufbau der IBAN

Die *IBAN* ist alphanumerisch. Die ersten beiden Stellen sind der Ländercode nach ISO 3166, z. B. *DE* für Deutschland. Danach folgt die zweistellige Prüfziffer, deren Berechnung im nächsten Abschnitt beschrieben wird. Es folgt die maximal 30stellige *BBAN (Basic Bank Account Number)*, die sich aus der *Institutsidentifikation (IID)* bzw. Bankleitzahl und der Kontonummer *(Bank Account Number, BAN)* zusammensetzt.[82]

Land	Länge	Struktur																											
		Land nach ISO		Prüf-ziffer		Basic Bank Account Number (BBAN)																							
Int.	max. 34	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Belgien	16	B	E	p	p	BLZ			Kontonummer																				
Dänemark	18	D	K	p	p	BLZ				Kontonummer																			
Deutschland	**22**	**D**	**E**	**p**	**p**	**BLZ**								**Kontonummer**															
Finnland	18	F	I	p	p	BLZ						Kontonummer																	
Frankreich	27	F	R	p	p	BLZ										Kontonummer													
Großbritannien	22	G	B	p	p	BLZ				Kontonummer																			
Irland	22	I	E	p	p	BLZ				Kontonummer																			
Island	26	I	S	p	p	BLZ				Kontonummer																			
Italien	27	I	T	p	p	BLZ											Kontonummer												
Luxemburg	20	L	U	p	p	BLZ				Kontonummer																			
Niederlande	18	N	L	p	p	BLZ				Kontonummer																			
Norwegen	15	N	O	p	p	BLZ				Kontonummer																			
Österreich	20	A	T	p	p	BLZ					Kontonummer																		
Polen	28	P	L	p	p	BLZ								Kontonummer															
Portugal	25	P	T	p	p	BLZ								Kontonummer															
Schweden	24	S	E	p	p	BLZ					Kontonummer																		
Schweiz	21	C	H	p	p	BLZ					Kontonummer																		
Spanien	24	E	S	p	p	BLZ								Kontonummer															

Abbildung 1: Aufbau der IBAN der europäischen Länder[83]

In Deutschland entspricht die achtstellige Bankleitzahl der IID und die zehnstellige Kontonummer der BAN. Somit hat die IBAN für eine deutsche Bankverbindung insgesamt 22 Stellen.

Das nachfolgende Beispiel zeigt praxisnah, wie die IBAN aussehen könnte.

---

81 Vgl. o.V., IBAN, Hrsg. Pruefziffernberechnung.DE, Stand: Juni 2002, http://www.pruefziffernberechnung.de/I/IBAN.shtml

82 Vgl. o.V., IBAN: International Bank Account Number, Hrsg. European Committee for Banking Standards, 1. Aufl., Brüssel 1996, S. 4

83 o.V., Pruefziffernberechnung.DE, a.a.O., S. 2

Beispiel: Ein Kunde der Sparkasse Heidelberg hat die Kontonummer 1234560 und möchte seine IBAN wissen.

Zuerst wird der Ländercode (ISO 3166) mit "00" ergänzt:

DE00

Dann folgt die achtstellige Bankleitzahl der Sparkasse Heidelberg:

67250020

Schließlich wird die Kontonummer zehnstellig hinzugefügt:

0001234560

Das Zwischenergebnis sieht so aus:

DE00672500200001234560

Da an Stelle der Prüfziffer noch die "00" steht, ist die Berechnung der Prüfziffer der nächste Schritt.

3.4.2 Berechnung der Prüfziffer

Zur Berechnung der Prüfziffer müssen zuerst die ersten vier Stellen der IBAN ans rechte Ende der Nummer verschoben werden.[84]

Beispiel: Die IBAN ohne Prüfziffer lautet:

DE00672500200001234560

Das Zwischenresultat sieht so aus:

67250020000123456 0**DE00**

Als Nächstes müssen alle Alphazeichen in numerische Zeichen umgewandelt werden. Dies geschieht mithilfe einer festgelegten Konversionstabelle:[85]

[84] Vgl. o.V., IBAN: International Bank Account Number, a.a.O., S. 5
[85] Vgl. o.V., IBAN: International Bank Account Number, a.a.O., S. 6

A = 10	G = 16	M = 22	S = 28	Y = 34
B = 11	H = 17	N = 23	T = 29	Z = 35
C = 12	I = 18	O = 24	U = 30	
D = 13	J = 19	P = 25	V = 31	
E = 14	K = 20	Q = 26	W = 32	
F = 15	L = 21	R = 27	X = 33	

Sonderzeichen und Leerstellen sind nicht zugelassen. Nach erfolgter Berechnung werden diese zweistelligen Zahlen in der IBAN wieder durch die ursprünglichen Alphazeichen ersetzt.

Beispiel: Nach Umwandlung der Buchstaben "DE" sieht das Zwischenresultat so aus:

67250020000123456**0131**400

Die so entstandene Zahl wird durch 97 geteilt und der Rest von 98 subtrahiert. Das Ergebnis ist die Prüfziffer. Ist das Ergebnis kleiner als 10, so wird eine führende 0 vorangestellt.[86]

Die zweistellige Prüfziffer wird anstelle der letzten beiden Nullen eingesetzt, die Alphastellen wieder an der ursprünglichen eingesetzt und der Block mit dem Ländercode und der Prüfziffer wieder an den Anfang gesetzt.

[86] Vgl. o.V., Prüfzifferberechnung, IBAN * IPI, Der neue Zahlungsbeleg, Hrsg. Swiss Interbank Clearing AG, Zürich 2000, http://www.telekurs-sic.ch, S. 6

Beispiel: Die obige Zahl wird durch 97 geteilt und der Rest ermittelt:

672500200001234560131400 : 97 = 6932991752590047011663 **Rest 89**

Der Rest wird von 98 subtrahiert:

98 – 89 = **09**

Die Prüfziffer wird eingesetzt:

67250020000123456013140**9**

Die Alphazeichen werden wieder eingesetzt:

67250020000123456**DE**09

Ländercode und Prüfziffer werden an den Anfang gestellt:

DE096725002000012345 60

Der Bankkunde kann nun mithilfe dieser IBAN am internationalen Zahlungsverkehr teilnehmen.

Da 97 die größte Primzahl unter 100 ist, bietet dieses Verfahren hervorragende Fehlererkennungseigenschaften. Als Prüfziffern sind Werte von 01 bis 98 möglich. Befinden sich an entsprechender Stelle in der Nummer nur Nullen, dann ist offensichtlich, dass die Prüfziffer noch nicht berechnet wurde.

Der Einsatz von Gewichtungsfaktoren ist beim Modulus-97-Verfahren zwar möglich, bringt jedoch nicht notwendigerweise eine bessere Fehlererkennung mit sich.

3.4.3 Kontrolle der IBAN-Prüfziffer

Zur Kontrolle der IBAN werden die eventuell vorangestellte Textkonstante "IBAN" und die Leerstellen entfernt. Wie auch zur Berechnung der Prüfziffer werden die ersten vier Stellen nach rechts verschoben und alle Alphazeichen gemäß der Konversionstabelle umgewandelt.

Die dadurch entstandene Zahl wird durch 97 geteilt. Bei korrekter Prüfziffer bleibt ein Divisionsrest von 1.

Beispiel: Die **IBAN DE09 6725 0020 0001 2345 60** soll geprüft werden.

Die Textkonstante und die Leerstellen werden entfernt:

DE09672500200001234560

Die ersten vier Stellen werden ans rechte Ende der Nummer verschoben:

67250020000123456 0**DE09**

Die Alphazeichen werden umgewandelt:

6725002000012345601**314**09

Die Zahl wird durch 97 geteilt:

6725002000012345601314 09 : 97 = 6932991752590047011664 **Rest 1**

Ergebnis: Die Nummer ist korrekt!

4 Zusammenfassung

4.1 Allgemeines

Ohne eine Benummerung der zu verwaltenden Objekte (Artikel, Konten, Verträge usw.) kommt die maschinelle Datenverarbeitung heute nicht mehr aus. Die korrekte Nummer ist der Schlüssel zu den Informationen über das Objekt.

Es muss entsprechend sichergestellt werden, dass stets die richtige Nummer verwendet wird. Eine falsche Nummer würde zu einer fehlerhaften Verarbeitung führen. Um dies zu verhindern muss an erster Stelle die Fehlervermeidung stehen. Organisatorische Maßnahmen halten die Fehlerquellen klein.

Darüber hinaus haben sich Fehlererkennungsmethoden durchgesetzt. Am verbreitetsten sind die Prüfziffernverfahren, die in der vorliegenden Arbeit ausführlich untersucht worden sind. Je nach Nummernsystem ist auch die Tabellenkontrolle eine sinnvolle Methode, Fehler zu erkennen. Ihr ausschließlicher Einsatz würde jedoch keine ausreichende Sicherheit gewährleisten. Kombinierte Schlüssel und das Multiplikationsverfahren haben sich nicht bewähren können.

Ob ein Fehlererkennungsverfahren eingesetzt wird, welches zur Anwendung kommt und wie aufwändig dieses sein darf, hängt nicht zuletzt von wirtschaftlichen Erwägungen ab. Die Kosten des Verfahrens müssen im Verhältnis zur Wahrscheinlichkeit und den Folgekosten eines Fehlers stehen.

Eine Nummer mit einer rechnerisch ermittelten Prüfziffer kann jederzeit ohne Zugriff auf die Stammdaten auf ihre Korrektheit überprüft werden. Da für heutige EDV-Systeme der Rechenaufwand eines Prüfziffernverfahrens keine große Rolle spielt, findet die Auswahl der geeigneten Verfahren vor allem in den verwendeten Nummernsystemen ihre Begrenzung. So sind beispielsweise Bankkontonummern in Deutschland immer 10stellig und numerisch. Mehrstellige Prüfziffern gehen zulasten der nutzbaren Stellen und nichtnumerische Prüfzeichen sind nicht möglich.

4.2 Kritik an heutigen Verfahren

Praktisch jedes Nummernsystem hat bereits eine Geschichte. So stammt etwa das *Einheitliche Kontonummernsystem der Sparkassen und Girozentralen (EKONS)* aus dem Jahr 1963. Bei Einführung der *Europäischen Artikelnummern (EAN)* wurde aus Kompatibilitätsgründen das Prüfziffernverfahren der Amerikanischen Artikelnummer UPC übernommen.

Die älteren verwendeten Prüfziffernverfahren sind leicht nachvollziehbar und können oft mit geringem Aufwand und auch ohne Unterstützung einer EDV-Anlage manuell gerechnet werden.

Mit Veränderungen in den Gewichtungsfaktoren können bestimmte Fehlerarten besser erkannt werden. Hierzu einige Beispiele:

- Die EAN verwendet ein Modulus-10-Verfahren mit den Gewichten 1 und 3. Zusätzliche Gewichte (7 und 9) könnten Doppeldrehfehler eliminieren.
- Das Modulus-10-Verfahren mit den Gewichten 1 und 2 bei laufender Quersummenbildung könnte verbessert werden, wenn die Gewichte 4, 5, 7 und 8 hinzu kämen.
- Beim Modulus-11-Verfahren ist eine Verbesserung möglich, wenn die Gewichte 5 und 6 nicht nebeneinander stehen. Die Gewichtungsfolge 2, 3, 4, 5, 7, 6, 8, 9, 10 wäre geeigneter.

Die Entscheidung für ein verbessertes Prüfziffernverfahren würde jedoch oft einen erheblichen Aufwand nach sich ziehen. Sparkassen und Banken müssen beispielsweise die Änderung bei der Bundesbank publizieren, ihre Kunden informieren und die Programme im Rechenzentrum umstellen.

Dieser Aufwand verursacht höhere Kosten, als der Nutzen des verbesserten Prüfziffernverfahrens sein könnte.

Kritische Erfahrungswerte und Verbesserungsvorschläge, die sich aus der Analyse ergeben haben, sind deshalb vor allem für die Neueinführung eines Nummernsystems von Bedeutung.

4.3 Welches Prüfziffernverfahren für welches Nummernsystem?

Häufig sehen Nummernsysteme von vornherein nur eine Stelle für die Prüfziffer vor. Oft ist zudem festgelegt, dass die Nummer nur Ziffern enthalten darf.

Sowohl das Modulus-10-Verfahren mit laufender Quersummenbildung als auch das Modulus-11-Verfahren bieten eine Fehlererkennungswahrscheinlichkeit von 98 %, wenn geeignete Gewichte (Stellenmultiplikatoren) verwendet werden.[87]

Das Modulus-11-Verfahren errechnet jedoch möglicherweise als Prüfziffer eine 10. In diesem Fall wird in der Regel entweder die entsprechende Nummer nicht vergeben

[87] Vgl. Jahnke, Bernd, a.a.O., S. 208

oder, wie in den modifizierten Verfahren, die 0 zweimal verwendet. Die Prüfziffer ist damit entweder eine *echte* oder eine *modifizierte* 0, was zulasten der Fehlererkennungswahrscheinlichkeit geht. Daher sind diese Verfahren nicht unbedingt empfehlenswert. Bei der ISBN hat sich hingegen mit dem X als römische Ziffer 10 eine sinnvolle Lösung durchgesetzt.

Beim Modulus-10-Verfahren können nicht alle Gewichtungsfaktoren verwendet werden. Entweder es werden die Gewichte 1, 3, 7 und 9 eingesetzt oder es kommen nach dem Verfahren mit laufender Quersummenbildung die Gewichte 1, 2, 4, 5, 7 und 8 infrage. Diese Einschränkung hat geringfügige mathematische Nachteile.

Die Restfehlerzahl ist bei den hier besprochenen Verfahren im Regelfall vertretbar oder kann durch andere Sicherungsmaßnahmen abgefangen werden.

Sind als Prüfziffer auch Buchstaben zugelassen, dann empfehlen sich das Modulus-23- oder das Modulus-26-Verfahren. Aufgrund ihres hohen Teilers ist die Restfehlerzahl geringer als bei den Modulus-10- und Modulus-11-Verfahren. Sie finden in der Praxis jedoch keine nennenswerte Verbreitung.

Bei zweistelligen Prüfziffern bieten die Modulus-97- und Modulus-101-Verfahren die besten Ergebnisse.[88] Das Modulus-101-Verfahren hat zwar eine geringere Restfehlerzahl, es könnte jedoch die Prüfziffer 100 errechnet werden. Analog dem Modulus-11-Verfahren ist dann die Nummer unbrauchbar oder es ist eine Modifikation vonnöten. Um diese Problematik zu umgehen, ist das Modulus-97-Verfahren eher zu empfehlen.

4.4 Ausblick

Die zunehmende Technisierung führt dazu, dass immer weniger Nummern manuell eingegeben werden. Statt des Personals in der Datenerfassung kommen Scanner und andere Lesegeräte zum Einsatz. Erübrigt sich hierdurch die Notwenigkeit eines Fehlererkennungsverfahrens?

Sicherlich nicht. Ein Schriftenlesesystem kann beispielsweise immer noch eine Ziffer falsch interpretieren. Sie kann auch ursächlich falsch hingeschrieben worden sein. Trotz neuerer Verfahren auf Bitebene werden die traditionellen Prüfziffernverfahren auf lange Zeit ihre Daseinsberechtigung nicht verlieren.

[88] Vgl. Jahnke, Bernd, a.a.O., S. 209

5 Programmbeschreibung

5.1 Allgemeines

Der beiliegende Datenträger enthält eine Datei namens PZV.xls. Zum Öffnen dieser Datei wird Microsoft Excel 97 oder höher benötigt.

Diese Datei beinhaltet

- ein Tabellenblatt mit der Schaltfläche *Prüfziffernverfahren,*
- eine benutzerdefinierte Symbolleiste,
- das Formular *frmPZV,*
- die zum Formular gehörenden Prozeduren,
- das Programmmodul *PZV* mit den benötigten Funktionen und Prozeduren.

Die Programmmodule sind mit *Microsoft Visual Basic 6.0* erstellt, wofür der in *Microsoft Office* integrierte *Visual Basic-Editor* verwendet wurde.

5.2 Aufruf des Programms über die Schaltfläche

Durch Anklicken der Schaltfläche *Prüfziffernverfahren* wird das Formular frmPZV angezeigt.

Der Benutzer wird aufgefordert, die zu berechnende bzw. zu prüfende Nummer einzugeben, das Prüfziffernberechnungsverfahren auszuwählen und schließlich die Befehlsschaltfläche mit der gewünschten Funktion zu betätigen.

Als Nummer schlägt das Formular den Wert vor, der sich in der aktuellen Zelle befindet.

Bei Betätigen der Schaltfläche *Prüfziffer berechnen* wird im Ergebnisfeld die Nummer mit angehängter Prüfziffer angezeigt, wie folgendes Beispiel zeigt.

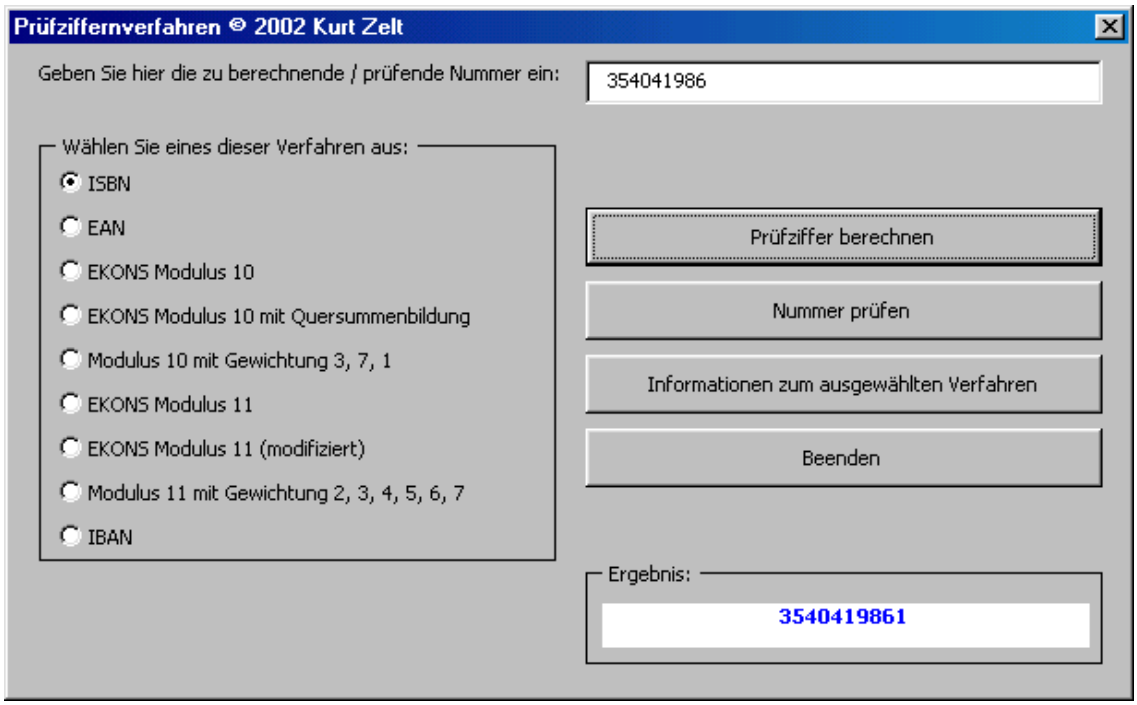

Abbildung 2: Formular frmPZV mit Rechenergebnis

Betätigt der Benutzer die Schaltfläche *Nummer prüfen*, dann wird die eingegebene Nummer nach der Rechenvorschrift des ausgewählten Verfahrens überprüft und als Ergebnis ausgegeben, dass die Nummer korrekt ist oder dass ein Fehler vorliegt.

In diesem Beispiel wurde eine ISBN auf ihre Richtigkeit getestet.

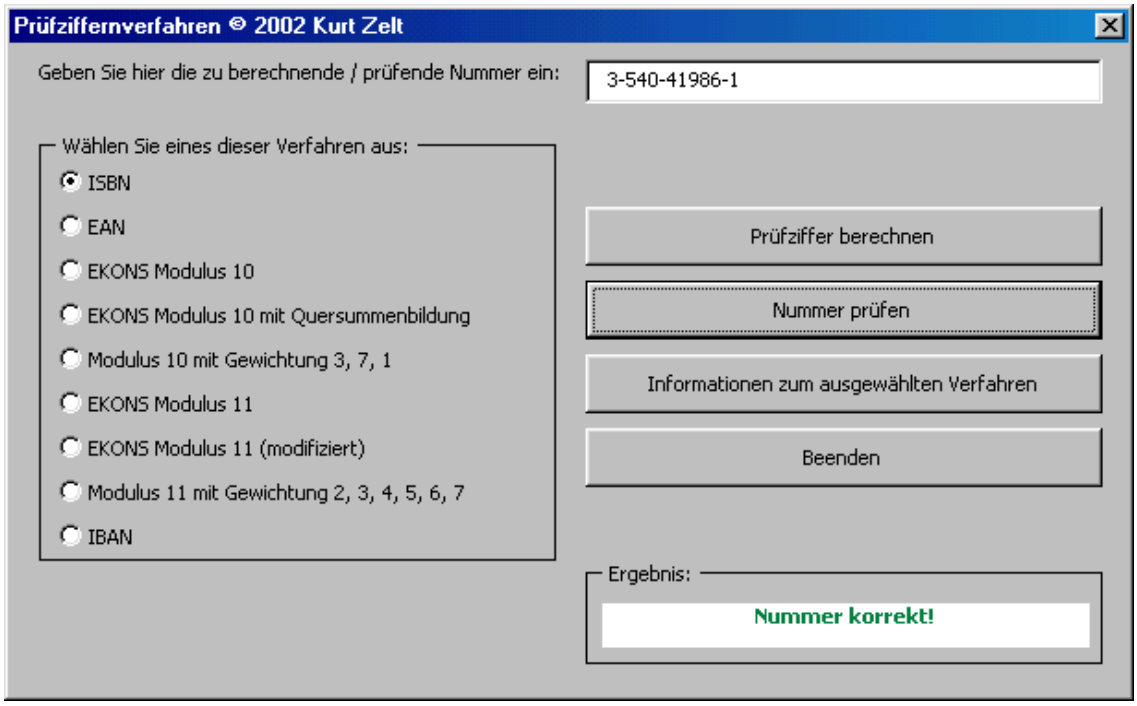

Abbildung 3: Formular frmPZV mit Prüfergebnis

Außerdem ist für jedes Verfahren eine kurze Beschreibung erhältlich. Bei Anklicken der Befehlsschaltfläche *Informationen zum ausgewählten Verfahren* öffnet sich ein Fenster mit den entsprechenden Erklärungen.

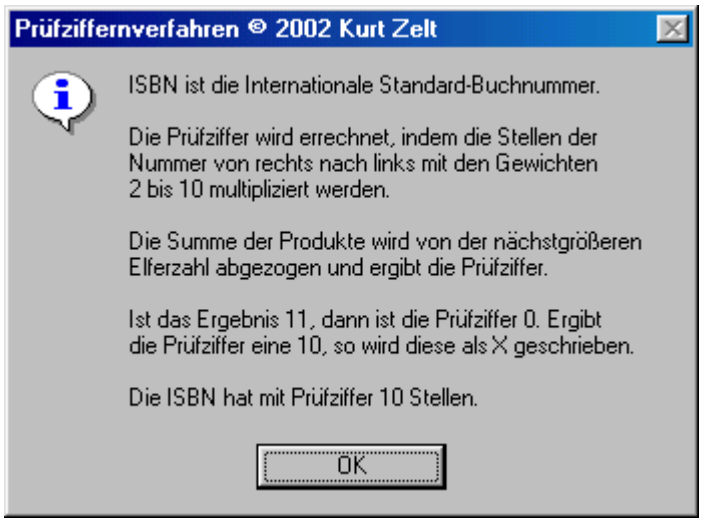

Abbildung 4: Infofenster

Beim Beenden wird der Benutzer gefragt, ob er den berechneten Wert in die aktuelle Zelle schreiben möchte. Ist das Ergebnis keine Nummer, sondern beispielsweise eine Fehlermeldung, dann erscheint dieses Fenster nicht.

Abbildung 5: Frage beim Beenden

5.3 Aufruf der Funktionen über die Symbolleiste

Alternativ zum oben beschriebenen Formular ermöglicht die benutzerdefinierte Symbolleiste *Prüfziffernverfahren* es dem Anwender, direkt auf die Rechenverfahren zuzugreifen. Hierzu ist die Markierung auf der Zelle zu positionieren, die die zu berechnende oder zu prüfende Zahl beinhaltet.

Nach Anklicken der Symbolfläche *Prüfziffernverfahren* öffnet sich die Auswahl der vorhandenen Rechenverfahren. Für jedes Verfahren stehen die Auswahlpunkte *Prüfziffer ermitteln*, *Nummer prüfen* und *Infos über...* zur Verfügung.

Wird *Prüfziffer ermitteln* ausgewählt, dann wird der Wert aus der aktuellen Zelle eingelesen, die Prüfziffer errechnet und die vollständige Nummer zurückgeschrieben.

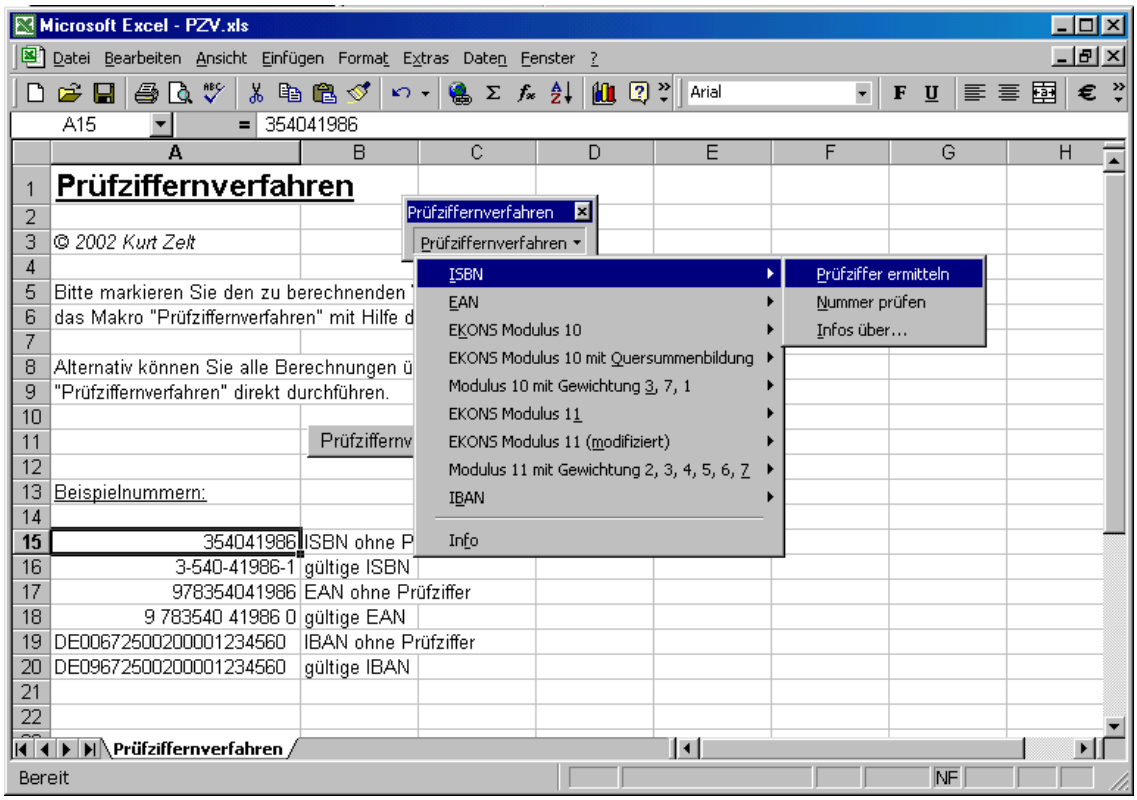

Abbildung 6: Start über die Symbolleiste

Trifft der Benutzer die Auswahl *Nummer prüfen*, dann wird die Nummer der aktuellen Zelle nach der Rechenvorschrift des ausgewählten Verfahrens überprüft. Das Ergebnis wird in einem Mitteilungsfenster ausgegeben.

Abbildung 7: Nummer korrekt oder nicht

Mithilfe der Auswahl *Infos über...* sind die Kurzbeschreibungen zu den jeweiligen Verfahren abrufbar.

5.4 Beschreibung des Programms

Das Programm besteht aus 63 Funktionen und Prozeduren, die im Modul *PZV* abgelegt sind und sechs weiteren Prozeduren, die dem Formular zugeordnet sind. Soweit die Programmroutinen bestimmten Prüfziffernverfahren zugeordnet sind, ist dies nach folgendem Schema ersichtlich:

Prüfziffernverfahren	**Bezeichnung in den Modulen**	**siehe Kapitel**
ISBN	ISBN	3.1
EAN	EAN	3.2
EKONS Modulus 10	EKONSm10	3.3.1
EKONS Modulus 10 mit Quersummen-bildung	EKONSm10Q	3.3.2
Modulus 10 mit Gewichtung 3, 7, 1	Mod10G371	3.3.3
EKONS Modulus 11	EKONSm11	3.3.4
EKONS Modulus 11 (modifiziert)	EKONSm11m	3.3.5
Modulus 11 mit Gewichtung 2, 3, 4, 5, 6, 7 (modifiziert)	Mod11G27	3.3.6
IBAN	IBAN	3.4

Die nachstehende Tabelle verschafft einen kurzen Überblick über die Aufgaben der einzelnen Funktionen und Prozeduren des Moduls *PZV*. Drei Punkte stehen für die jeweiligen Prüfziffernverfahren.

Modul	**Funktion**	**erwartet Wert**	**gibt zurück**
Start	öffnet das Formular		
MenüErrechne_...	wird aus der Symbolleiste aufgerufen, ruft Errechne_... auf		
MenüPrüfe_...	wird aus der Symbolleiste aufgerufen, ruft Prüfe_... auf		
Errechne_...	prüft die Eingabe und ruft PZ_... auf	String	String
Prüfe_...	prüft die Eingabe und prüft die Nummer nach der Rechenvorschrift	String	Boolean
Info	gibt Infos über die Arbeit aus		
Info_...	gibt Infos über das jeweilige Verfahren aus		
PZ_...	berechnet die Prüfziffer nach der jeweiligen Rechenvorschrift	String	String
ClearIBAN	entfernt nicht notwenige Zeichen aus einer IBAN	String	String
IsIBAN	prüft Länderzeichen und Länge einer IBAN	String	Boolean
Numerisch	entfernt nicht numerische Zeichen	String	String
Quersumme	ermittelt die Quersumme aus einer Zahl	Integer	Integer
IBAN2Z	macht aus einer IBAN eine Zahl	String	String
Rest97	ermittelt den Divisionsrest	String	Integer
Fehler	gibt eine Fehlermeldung aus	Integer	

5.5 Fehlerbehandlung

Da gleiche Fehler in unterschiedlichen Programmmodulen festgestellt werden könnten, wurde die Fehlerausgabe in ein separates Unterprogramm ausgelagert. Dieses wird bei Bedarf aufgerufen und eine Fehlernummer übergeben. Bei Fehlern in der Nummernlänge besteht zusätzlich die Möglichkeit, die Anzahl der erwarteten Zeichen anzugeben.

Im Einzelnen sind folgende Fehlerarten berücksichtigt:

- Die eingegebene Nummer ist zu lang (bei fester Länge)
- Die eingegebene Nummer ist zu kurz (bei fester Länge)
- Die eingegebene Nummer ist zu kurz (Mindestlänge)
- Die EAN wurde mit falscher Stellenanzahl eingegeben (mit / ohne Prüfziffer)
- Es wurde keine Berechnungsmethode ausgewählt
- Die Eingabe enthält keine Ziffern
- Es wurde keine gültige IBAN eingegeben
- Die Prüfziffer der IBAN ist nicht 00 / ist 00

Die Fehlermeldung wird in einem Mitteilungsfenster ausgegeben, das Programm jedoch fortgesetzt. Eine zu prüfende Nummer wird automatisch als falsch eingestuft, eine zu berechnende Nummer zeigt die Fehlerbezeichnung als Ergebnis an. Wurde die Berechnung über den Symbolleistenbefehl ausgelöst, dann wird der Wert in der aktuellen Zelle nicht verändert.

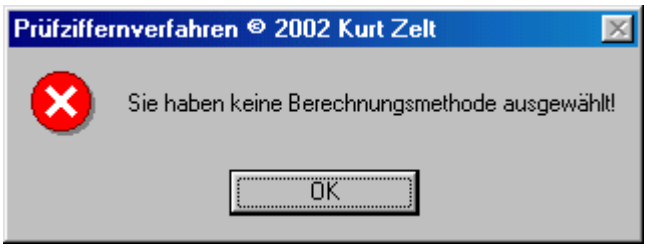

Abbildung 8: Beispiel einer Fehlermeldung

Anhang

Quellcode des Hauptprogramms

```
' Programm: Prüfziffernverfahren
' Funktion: Ermitteln und Prüfen von Prüfziffern
' Autor:    Kurt Zelt
' Stand:    September 2002

Option Explicit
Sub Start()
    '
    ' Formular öffnen;
    ' alle weiteren Aktionen werden über das Formular gestartet
    '
    frmPZV.Show
End Sub


'
' Prozeduren, die über ein Menü aufgerufen werden können
'


Sub MenüErrechne_ISBN()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_ISBN(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_EAN()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_EAN(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_EKONSm10()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_EKONSm10(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_EKONSm10Q()
```

```
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_EKONSm10Q(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_Mod10G371()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_Mod10G371(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_EKONSm11()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_EKONSm11(Eingabe)
    If Left(Ausgabe, 3) = "Nr." Then
        MsgBox Ausgabe, vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    ElseIf Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_EKONSm11m()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_EKONSm11m(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüErrechne_Mod11G27()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_Mod11G27(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub
```

```
Sub MenüErrechne_IBAN()
    '
    ' Hole Wert aus aktiver Zelle, berechne Prüfziffer, schreibe zurück
    '
    Dim Eingabe As String
    Dim Ausgabe As String
    Eingabe = ActiveCell.Value
    Ausgabe = Errechne_IBAN(Eingabe)
    If Left(Ausgabe, 6) <> "Fehler" Then
        ActiveCell.Value = Ausgabe
    End If
End Sub


Sub MenüPrüfe_ISBN()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_ISBN(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_EAN()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_EAN(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_EKONSm10()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_EKONSm10(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub
```

```
Sub MenüPrüfe_EKONSm10Q()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_EKONSm10Q(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_Mod10G371()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_Mod10G371(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_EKONSm11()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_EKONSm11(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_EKONSm11m()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_EKONSm11m(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub
```

```
Sub MenüPrüfe_Mod11G27()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_Mod11G27(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


Sub MenüPrüfe_IBAN()
    '
    ' Hole Wert aus aktiver Zelle, prüfe Nummer, zeige Ergebnis an
    '
    Dim Eingabe As String
    Dim Ergebnis As Boolean
    Eingabe = ActiveCell.Value
    Ergebnis = Prüfe_IBAN(Eingabe)
    If Ergebnis Then
        MsgBox "Nummer korrekt!", vbInformation, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Else
        MsgBox "Fehler!", vbCritical, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    End If
End Sub


'
' Funktionen zum Errechnen, Prüfen und Anzeigen der Infos
'


Function Errechne_ISBN(Eingabe As String) As String
    '
    ' Prüfen der eingegebenen ISBN auf ihre Gültigkeit;
    ' Aufruf des Unterprogrammes zum Errechnen der Prüfziffer;
    ' Rückgabe der vervollständigten ISBN
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_ISBN = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_ISBN = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 9 Then
        Call Fehler(2, 9)
        Errechne_ISBN = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_ISBN(Nummer)
```

```
        '
        ' Ergebnis zurückgeben
        '
        Errechne_ISBN = Nummer & PZ
    End If
End Function


Function Errechne_EAN(Eingabe As String) As String
    '
    ' Prüfen der eingegebenen EAN auf ihre Gültigkeit;
    ' Aufruf des Unterprogrammes zum Errechnen der Prüfziffer;
    ' Rückgabe der vervollständigten EAN
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Len(Nummer) = 7 Or Len(Nummer) = 12 Then
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_EAN(Nummer)
        '
        ' Ergebnis zurückgeben
        '
        Errechne_EAN = Nummer & PZ
    Else
        Call Fehler(5)
        Errechne_EAN = "Fehler: Falsche Länge der Nummer"
    End If
End Function


Function Errechne_EKONSm10(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_EKONSm10 = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_EKONSm10 = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
        Errechne_EKONSm10 = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_EKONSm10(Nummer)
        '
        ' Ergebnis zurückgeben
```

```
        '
        Errechne_EKONSm10 = Nummer & PZ
    End If
End Function


Function Errechne_EKONSm10Q(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_EKONSm10Q = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_EKONSm10Q = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
        Errechne_EKONSm10Q = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_EKONSm10Q(Nummer)
        '
        ' Ergebnis zurückgeben
        '
        Errechne_EKONSm10Q = Nummer & PZ
    End If
End Function


Function Errechne_Mod10G371(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_Mod10G371 = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_Mod10G371 = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
```

```
        Errechne_Mod10G371 = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_Mod10G371(Nummer)
        '
        ' Ergebnis zurückgeben
        '
        Errechne_Mod10G371 = Nummer & PZ
    End If
End Function


Function Errechne_EKONSm11(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_EKONSm11 = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_EKONSm11 = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
        Errechne_EKONSm11 = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_EKONSm11(Nummer)
        If Left(PZ, 1) = "N" Then
            '
            ' Nummer kann nicht vergeben werden
            '
            Errechne_EKONSm11 = PZ
        Else
            '
            ' Ergebnis zurückgeben
            '
            Errechne_EKONSm11 = Nummer & PZ
        End If
    End If
End Function


Function Errechne_EKONSm11m(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
```

```
    ' Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_EKONSm11m = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_EKONSm11m = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
        Errechne_EKONSm11m = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_EKONSm11m(Nummer)
        '
        ' Ergebnis zurückgeben
        '
        Errechne_EKONSm11m = Nummer & PZ
    End If
End Function


Function Errechne_Mod11G27(Eingabe As String) As String
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der Kontonummer
    '
    Dim Nummer As String
    Dim PZ As String
    '
    ' Gliederungszeichen entfernen;
    ' Prüfen, ob Eingabe gültig
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Errechne_Mod11G27 = ""
    ElseIf Len(Nummer) > 9 Then
        Call Fehler(1, 9)
        Errechne_Mod11G27 = "Fehler: Nummer zu groß"
    ElseIf Len(Nummer) < 3 Then
        '
        ' Kontonummern mit weniger als 3 Stellen kommen
        ' in der Praxis nicht vor
        '
        Call Fehler(4, 3)
        Errechne_Mod11G27 = "Fehler: Nummer zu klein"
    Else
        '
        ' Prüfziffer ermitteln
        '
        PZ = PZ_Mod11G27(Nummer)
        '
        ' Ergebnis zurückgeben
        '
        Errechne_Mod11G27 = Nummer & PZ
    End If
End Function


Function Errechne_IBAN(Eingabe As String) As String
    '
```

```
    ' Prüfen der eingegebenen IBAN auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Rückgabe der IBAN mit Prüfziffer
    '
    Dim Gültig As Boolean
    Dim IBAN As String
    Dim I As Integer       ' I = Zähler für Iteration
    Dim Z As String        ' Z = (ein) Zeichen
    Dim PZ As String
    '
    ' Leerstellen, Gliederungszeichen usw. entfernen
    '
    IBAN = ClearIBAN(Eingabe)
    '
    ' Hat die IBAN eine gültige Länge?
    '
    Gültig = IsIBAN(IBAN)
    If Gültig = True Then
        '
        ' Ist die Prüfziffer mit "00" vorbelegt?
        '
        If Mid(IBAN, 3, 2) = "00" Then
            PZ = PZ_IBAN(IBAN)
            '
            ' die Prüfziffer kommt an die 3. und 4. Stelle
            '
            Errechne_IBAN = Left(IBAN, 2) & PZ _
                & Right(Eingabe, Len(IBAN) - 4)
        Else
            '
            ' Die Prüfziffer ist nicht "00"
            '
            Call Fehler(11)
            Errechne_IBAN = "Fehler: Prüfziffer belegt"
        End If
    Else
        '
        ' Es wurde keine gültige IBAN eingegeben
        '
        Call Fehler(10)
        Errechne_IBAN = "Fehler: Keine gültige IBAN"
    End If
End Function


Function Prüfe_ISBN(Eingabe As String) As Boolean
    '
    ' Prüfen der eingegebenen ISBN auf Gültigkeit;
    ' Unterprogramm zum Errechnen der Prüfziffer aufrufen;
    ' Vergleich der errechneten mit der vorhandenen Prüfziffer
    '
    Dim Nummer As String
    Dim PZ_Eingabe As String
    Dim PZ_Errechnet As String
    '
    ' Gliederungszeichen entfernen;
    ' Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Right(Eingabe, 1) = "x" Or Right(Eingabe, 1) = "X" Then
        '
        ' falls die Prüfziffer ein X war, muss sie hier wieder
        ' angehängt werden, da die Funktion Numerisch dieses abschneidet
        '
        Nummer = Nummer & "X"
    End If
    If Nummer = "" Then
        Call Fehler(9)
        Prüfe_ISBN = False
    ElseIf Len(Nummer) > 10 Then
```

```
        Call Fehler(1, 10)
        Prüfe_ISBN = False
    ElseIf Len(Nummer) < 10 Then
        Call Fehler(2, 10)
        Prüfe_ISBN = False
    Else
        '
        ' Prüfziffer ermitteln;
        ' hierzu muss die Nummer ohne Prüfziffer weitergegeben werden
        '
        PZ_Eingabe = Right(Nummer, 1)
        Nummer = Left(Nummer, 9)
        PZ_Errechnet = PZ_ISBN(Nummer)
        '
        ' Ergebnis mit der Eingabe vergleichen
        '
        Prüfe_ISBN = (PZ_Eingabe = PZ_Errechnet)
    End If
End Function


Function Prüfe_EAN(Eingabe As String) As Boolean
    '
    ' Prüfen der eingegebenen EAN auf ihre Gültigkeit;
    ' Durchführung der Kontrolle nach der Rechenvorschrift,
    ' nach der nach der Division kein Rest bleiben darf
    '
    Dim Nummer As String
    Dim L As Integer     ' L = Länge der Nummer
    Dim I As Integer     ' I = Zähler für die Iteration
    Dim S As Long        ' S = Summe der gewichteten Ziffern
    S = 0
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    L = Len(Nummer)
    If L = 8 Or L = 13 Then
        '
        ' die ungeraden Stellen (von rechts) sind mit 3 gewichtet
        '
        For I = 1 To L - 1 Step 2
            S = S + Mid(Nummer, L - I, 1) * 3
        Next I
        '
        ' die geraden Stellen (von rechts) sind mit 1 gewichtet,
        ' die Prüfziffer dazuaddieren
        '
        For I = 0 To L - 1 Step 2
            S = S + Mid(Nummer, L - I, 1)
        Next I
        '
        ' die Prüfziffer ist korrekt, wenn die Summe ohne Rest
        ' durch 10 teilbar ist
        '
        Prüfe_EAN = (S Mod 10 = 0)
    ElseIf Nummer = "" Then
        Call Fehler(9)
        Prüfe_EAN = False
    Else
        Call Fehler(6)
        Prüfe_EAN = False
    End If
End Function


Function Prüfe_EKONSm10(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf Gültigkeit;
```

```
    ' Durchführen der Kontrolle nach der Rechenvorschrift,
    ' nach der nach der Division kein Rest bleiben darf
    '
    Dim Nummer As String
    Dim L As Integer     ' L = Länge der Nummer
    Dim I As Integer     ' I = Zähler für die Iteration
    Dim S As Long        ' S = Summe der gewichteten Ziffern
    S = 0
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    L = Len(Nummer)
    If L = 0 Then
        Call Fehler(9)
        Prüfe_EKONSm10 = False
    ElseIf L > 10 Then
        Call Fehler(1, 10)
        Prüfe_EKONSm10 = False
    ElseIf L < 3 Then
        Call Fehler(4, 3)
        Prüfe_EKONSm10 = False
    Else
        '
        ' die ungeraden Stellen (von rechts) sind mit 2 gewichtet
        '
        For I = 1 To L - 1 Step 2
            S = S + Mid(Nummer, L - I, 1) * 2
        Next I
        '
        ' die geraden Stellen (von rechts) sind mit 1 gewichtet,
        ' die Prüfziffer dazuaddieren
        '
        For I = 0 To L - 1 Step 2
            S = S + Mid(Nummer, L - I, 1)
        Next I
        '
        ' die Prüfziffer ist korrekt, wenn die Summe ohne Rest
        ' durch 10 teilbar ist
        '
        Prüfe_EKONSm10 = (S Mod 10 = 0)
    End If
End Function


Function Prüfe_EKONSm10Q(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Durchführen der Kontrolle nach der Rechenvorschrift,
    ' nach der nach der Division kein Rest bleiben darf
    '
    Dim Nummer As String
    Dim L As Integer     ' L = Länge der Nummer
    Dim I As Integer     ' I = Zähler für die Iteration
    Dim S As Long        ' S = Summe der gewichteten Ziffern
    S = 0
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    L = Len(Nummer)
    If L = 0 Then
        Call Fehler(9)
        Prüfe_EKONSm10Q = False
    ElseIf L > 10 Then
        Call Fehler(1, 10)
        Prüfe_EKONSm10Q = False
    ElseIf L < 3 Then
```

```
        Call Fehler(4, 3)
        Prüfe_EKONSm10Q = False
    Else
        '
        ' die ungeraden Stellen (von rechts) sind mit 2 gewichtet;
        ' aus zweistelligen Ergebnissen die Quersumme nehmen
        '
        For I = 1 To L - 1 Step 2
            S = S + Quersumme(Mid(Nummer, L - I, 1) * 2)
        Next I
        '
        ' die geraden Stellen (von rechts) sind mit 1 gewichtet,
        ' die Prüfziffer dazuaddieren
        '
        For I = 0 To L - 1 Step 2
            S = S + Mid(Nummer, L - I, 1)
        Next I
        '
        ' die Prüfziffer ist korrekt, wenn die Summe ohne Rest
        ' durch 10 teilbar ist
        '
        Prüfe_EKONSm10Q = (S Mod 10 = 0)
    End If
End Function


Function Prüfe_Mod10G371(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Durchführen der Kontrolle nach der Rechenvorschrift,
    ' nach der nach der Division kein Rest bleibt
    '
    Dim Nummer As String
    Dim L As Integer     ' L = Länge der Nummer
    Dim I As Integer     ' I = Zähler für die Iteration
    Dim S As Long        ' S = Summe der gewichteten Ziffern
    S = 0
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    L = Len(Nummer)
    If L = 0 Then
        Call Fehler(9)
        Prüfe_Mod10G371 = False
    ElseIf L > 10 Then
        Call Fehler(1, 10)
        Prüfe_Mod10G371 = False
    ElseIf L < 3 Then
        Call Fehler(4, 3)
        Prüfe_Mod10G371 = False
    Else
        '
        ' die erste, vierte, siebte usw. Stelle von rechts
        ' wird mit 3 gewichtet
        ' (Achtung: ganz rechts steht jetzt die PZ)
        '
        For I = 1 To L - 1 Step 3
            S = S + Mid(Nummer, L - I, 1) * 3
        Next I
        '
        ' die zweite, fünfte, achte usw. Stelle von rechts
        ' wird mit 7 gewichtet
        '
        For I = 2 To L - 1 Step 3
            S = S + Mid(Nummer, L - I, 1) * 7
        Next I
        '
        ' die dritte, sechste, neunte usw. Stelle von rechts
```

```
        ' wird mit 1 gewichtet
        ' (und die PZ, die ganz rechts steht)
        '
        For I = 0 To L - 1 Step 3
            S = S + Mid(Nummer, L - I, 1)
        Next I
        '
        ' die Prüfziffer ist korrekt, wenn die Summe ohne Rest
        ' durch 10 teilbar ist
        '
        Prüfe_Mod10G371 = (S Mod 10 = 0)
    End If
End Function


Function Prüfe_EKONSm11(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Aufruf des Unterprogramms zum Errechnen der Prüfziffer;
    ' Vergleich des errechneten mit dem vorhandenen Zeichen
    '
    Dim Nummer As String
    Dim PZ_Eingabe As String
    Dim PZ_Errechnet As String
    '
    ' Gliederungszeichen entfernen,
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Prüfe_EKONSm11 = False
    ElseIf Len(Nummer) > 10 Then
        Call Fehler(1, 10)
        Prüfe_EKONSm11 = False
    ElseIf Len(Nummer) < 3 Then
        Call Fehler(4, 3)
        Prüfe_EKONSm11 = False
    Else
        '
        ' Prüfziffer ermitteln,
        ' hierzu die Nummer ohne Prüfziffer weitergeben
        '
        PZ_Eingabe = Right(Nummer, 1)
        Nummer = Left(Nummer, Len(Nummer) - 1)
        PZ_Errechnet = PZ_EKONSm11(Nummer)
        '
        ' das Ergebnis mit der Eingabe vergleichen
        '
        Prüfe_EKONSm11 = (PZ_Eingabe = PZ_Errechnet)
    End If
End Function


Function Prüfe_EKONSm11m(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf Gültigkeit;
    ' Aufruf des Unterprogramms zum Errechnen der Prüfziffer;
    ' Vergleich des errechneten mit dem vorhandenen Zeichen
    '
    Dim Nummer As String
    Dim PZ_Eingabe As String
    Dim PZ_Errechnet As String
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
```

```
        Prüfe_EKONSm11m = False
    ElseIf Len(Nummer) > 10 Then
        Call Fehler(1, 10)
        Prüfe_EKONSm11m = False
    ElseIf Len(Nummer) < 3 Then
        Call Fehler(4, 3)
        Prüfe_EKONSm11m = False
    Else
        '
        ' Prüfziffer ermitteln;
        ' hierzu die Nummer ohne Prüfziffer weitergeben
        '
        PZ_Eingabe = Right(Nummer, 1)
        Nummer = Left(Nummer, Len(Nummer) - 1)
        PZ_Errechnet = PZ_EKONSm11m(Nummer)
        '
        ' Ergebnis mit der Eingabe vergleichen
        '
        Prüfe_EKONSm11m = (PZ_Eingabe = PZ_Errechnet)
    End If
End Function


Function Prüfe_Mod11G27(Eingabe As String) As Boolean
    '
    ' Prüfen der Eingabe auf ihre Gültigkeit;
    ' Aufruf des Unterprogramms zum Errechnen der Prüfziffer auf;
    ' Vergleich des errechneten mit dem vorhandenen Zeichen
    '
    Dim Nummer As String
    Dim PZ_Eingabe As String
    Dim PZ_Errechnet As String
    '
    ' Gliederungszeichen entfernen;
    ' dann Prüfen, ob die Eingabe gültig ist
    '
    Nummer = Numerisch(Eingabe)
    If Nummer = "" Then
        Call Fehler(9)
        Prüfe_Mod11G27 = False
    ElseIf Len(Nummer) > 10 Then
        Call Fehler(1, 10)
        Prüfe_Mod11G27 = False
    ElseIf Len(Nummer) < 3 Then
        Call Fehler(4, 3)
        Prüfe_Mod11G27 = False
    Else
        '
        ' Prüfziffer ermitteln;
        ' hierzu die Nummer ohne Prüfziffer weitergeben
        '
        PZ_Eingabe = Right(Nummer, 1)
        Nummer = Left(Nummer, Len(Nummer) - 1)
        PZ_Errechnet = PZ_Mod11G27(Nummer)
        '
        ' Ergebnis wird mit der Eingabe vergleichen
        '
        Prüfe_Mod11G27 = (PZ_Eingabe = PZ_Errechnet)
    End If
End Function


Function Prüfe_IBAN(Eingabe As String) As Boolean
    '
    ' Prüfen der eingegebenen IBAN auf ihre Gültigkeit;
    ' Durchführen der Kontrolle nach der Rechenvorschrift,
    ' nach der nach der Division der Rest 1 sein muss
    '
    Dim Gültig As Boolean
    Dim IBAN As String
```

```
    Dim Zahl As String
    Dim I As Integer       ' I = Zähler für Iteration
    Dim Z As String        ' Z = (ein) Zeichen
    '
    ' Leerstellen, Gliederungszeichen usw. entfernen
    '
    IBAN = ClearIBAN(Eingabe)
    '
    ' Hat die IBAN eine gültige Länge?
    '
    Gültig = IsIBAN(IBAN)
    If Gültig = True Then
        '
        ' Ist die Prüfziffer "00"?
        '
        If Mid(IBAN, 3, 2) = "00" Then
            Call Fehler(12)
            Prüfe_IBAN = False
        Else
            '
            ' Aus der IBAN eine Zahl machen
            '
            Zahl = IBAN2Z(IBAN)
            '
            ' Zahl durch 97 teilen, Rest muss 1 sein
            '
            Prüfe_IBAN = (Rest97(Zahl) = 1)
        End If
    Else
        '
        ' Es wurde keine gültige IBAN eingegeben
        '
        Call Fehler(10)
        Prüfe_IBAN = False
    End If
End Function


Sub Info()
    '
    ' Allgemeine Infos
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "Prüfziffernverfahren" & vbCr & vbCr
    A = A & "Diplomarbeit für den Studiengang" & vbCr
    A = A & "Betriebswirt (VWA) - Schwerpunkt Informationsmanagement"
    A = A & vbCr & vbCr & "Prüfer:" & vbCr
    A = A & "Dipl.-Math. Edith Mechelke-Schwede, Viernheim" & vbCr
    A = A & "Verwaltungs- und Wirtschafts-Akademie Rhein-Neckar e. V."
    A = A & ", Mannheim     " & vbCr & vbCr
    A = A & "Eingereicht von:" & vbCr
    A = A & "Kurt Zelt, Wilhelmstr. 1, 68723 Plankstadt" & vbCr
    A = A & "tel. 06202/924723, www.zelt-online.de" & vbCr
    A = A & "Stand: September 2002" & vbCr & vbCr
    A = A & "Dieses Programm ist urheberrechtlich geschützt." & vbCr
    A = A & "Weitergabe nur mit Zustimmung des Autors."
    MsgBox A, vbInformation, T
End Sub


Sub Info_ISBN()
    '
    ' Die Information über die ISBN als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "ISBN ist die Internationale Standard-Buchnummer." & vbCr & vbCr
```

```
    A = A & "Die Prüfziffer wird errechnet, indem die Stellen der" & vbCr
    A = A & "Nummer von rechts nach links mit den Gewichten" & vbCr
    A = A & "2 bis 10 multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächstgrößeren     " & vbCr
    A = A & "Elferzahl abgezogen und ergibt die Prüfziffer." & vbCr & vbCr
    A = A & "Ist das Ergebnis 11, dann ist die Prüfziffer 0. Ergibt" & vbCr
    A = A & "die Prüfziffer eine 10, so wird diese als X geschrieben." _
        & vbCr & vbCr
    A = A & "Die ISBN hat mit Prüfziffer 10 Stellen."
    MsgBox A, vbInformation, T
End Sub


Sub Info_EAN()
    '
    ' Die Information über die EAN als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "Die EAN ist die Internationale Artikelnummer," & vbCr
    A = A & "ursprünglich Europäische Artikelnummer." & vbCr & vbCr
    A = A & "Die Prüfziffer wird errechnet, indem die Stellen" & vbCr
    A = A & "der Nummer von rechts nach links mit den" & vbCr
    A = A & "Gewichten 3 und 1 multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe hieraus wird von der nächstgrößeren     " & vbCr
    A = A & "Zehnerzahl abgezogen und ergibt die Prüfziffer." & vbCr
    A = A & "Ist das Ergebnis 10, dann ist die Prüfziffer 0." & vbCr & vbCr
    A = A & "Die EAN hat mit Prüfziffer 13 Stellen." & vbCr
    A = A & "Die Kurzversion hat 8 Stellen."
    MsgBox A, vbInformation, T
End Sub


Sub Info_EKONSm10()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "EKONS ist das Einheitliche Kontonummernsystem" & vbCr
    A = A & "der Sparkassen und Girozentralen." & vbCr & vbCr
    A = A & "Dieses Kontonummernsystem lässt die Berechnung     " & vbCr
    A = A & "der Prüfziffer nach verschiedenen Varianten zu." & vbCr & vbCr
    A = A & "Beim Modulus-10-Verfahren wird die Prüfziffer" & vbCr
    A = A & "errechnet, indem die Stellen der Nummer von" & vbCr
    A = A & "rechts nach links mit den Gewichten 2 und 1" & vbCr
    A = A & "multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Zehnerzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer. Ist das Ergebnis 10, dann ist die Prüf-" & vbCr
    A = A & "ziffer 0." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_EKONSm10Q()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "EKONS ist das Einheitliche Kontonummernsystem" & vbCr
    A = A & "der Sparkassen und Girozentralen." & vbCr & vbCr
    A = A & "Dieses Kontonummernsystem lässt die Berechnung     " & vbCr
    A = A & "der Prüfziffer nach verschiedenen Varianten zu." & vbCr & vbCr
```

```
    A = A & "Beim Modulus-10-Verfahren mit laufender Quer-" & vbCr
    A = A & "summenbildung wird die Prüfziffer errechnet," & vbCr
    A = A & "indem die Stellen der Nummer von rechts nach" & vbCr
    A = A & "links mit den Gewichten 2 und 1 multipliziert" & vbCr
    A = A & "werden. Aus zweistelligen Produkten wird die" & vbCr
    A = A & "Quersumme gebildet." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Zehnerzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer. Ist das Ergebnis 10, dann ist die Prüf-" & vbCr
    A = A & "ziffer 0." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_Mod10G371()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "Modulus 10, Gewichtung 3, 7, 1" & vbCr & vbCr
    A = A & "Dieses Prüfziffernverfahren nutzen hauptsächlich die     " & vbCr
    A = A & "württembergischen Sparkassen zur Berechnung der" & vbCr
    A = A & "Prüfziffern für ihre Kontonummern." & vbCr & vbCr
    A = A & "Die Prüfziffer wird errechnet, indem die Stellen der" & vbCr
    A = A & "Nummer von rechts nach links mit den Gewichten" & vbCr
    A = A & "3, 7 und 1 multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Zehnerzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer. Ist das Ergebnis 10, dann ist die Prüf-" & vbCr
    A = A & "ziffer 0." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_EKONSm11()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "EKONS ist das Einheitliche Kontonummernsystem" & vbCr
    A = A & "der Sparkassen und Girozentralen." & vbCr & vbCr
    A = A & "Dieses Kontonummernsystem lässt die Berechnung     " & vbCr
    A = A & "der Prüfziffer nach verschiedenen Varianten zu." & vbCr & vbCr
    A = A & "Beim Modulus-11-Verfahren wird die Prüfziffer" & vbCr
    A = A & "errechnet, indem die Stellen der Nummer von" & vbCr
    A = A & "rechts nach links mit den Gewichten 2 bis 10" & vbCr
    A = A & "multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Elferzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer." & vbCr & vbCr
    A = A & "Ist das Ergebnis 11, dann ist die Prüfziffer 0. Ergibt" & vbCr
    A = A & "die Prüfziffer eine 10, so ist die Nummer nicht" & vbCr
    A = A & "verwendbar." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_EKONSm11m()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
```

```
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "EKONS ist das Einheitliche Kontonummernsystem" & vbCr
    A = A & "der Sparkassen und Girozentralen." & vbCr & vbCr
    A = A & "Dieses Kontonummernsystem lässt die Berechnung     " & vbCr
    A = A & "der Prüfziffer nach verschiedenen Varianten zu." & vbCr & vbCr
    A = A & "Beim modifizierten Modulus-11-Verfahren wird die" & vbCr
    A = A & "Prüfziffer errechnet, indem die Stellen der Nummer" & vbCr
    A = A & "von rechts nach links mit den Gewichten 2 bis 10" & vbCr
    A = A & "multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Elferzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer." & vbCr & vbCr
    A = A & "Ist das Ergebnis 10 oder 11, dann ist die Prüfziffer" & vbCr
    A = A & "in beiden Fällen 0." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_Mod11G27()
    '
    ' Die Information über dieses Verfahren als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "Modulus 11, Gewichtung 2, 3, 4, 5, 6, 7" & vbCr & vbCr
    A = A & "Dieses Prüfziffernverfahren ist hauptsächlich bei" & vbCr
    A = A & "Volksbanken im Einsatz, aber auch z. B. bei der" & vbCr
    A = A & "Sparkasse Heidelberg." & vbCr & vbCr
    A = A & "Die Prüfziffer wird errechnet, indem die Stellen der     " & vbCr
    A = A & "Nummer von rechts nach links mit den Gewichten" & vbCr
    A = A & "2, 3, 4, 5, 6, 7 multipliziert werden." & vbCr & vbCr
    A = A & "Die Summe der Produkte wird von der nächst-" & vbCr
    A = A & "größeren Elferzahl abgezogen und ergibt die" & vbCr
    A = A & "Prüfziffer." & vbCr & vbCr
    A = A & "Ist das Ergebnis 10 oder 11, dann ist die Prüfziffer" & vbCr
    A = A & "in beiden Fällen 0." & vbCr & vbCr
    A = A & "Die Kontonummer hat 9 Stellen plus die Prüfziffer." & vbCr
    A = A & "Kürzere Nummern sind möglich."
    MsgBox A, vbInformation, T
End Sub


Sub Info_IBAN()
    '
    ' Die Information über die IBAN als Message-Box ausgeben
    '
    Dim A As String
    Dim T As String
    T = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    A = "Die IBAN (International Bank Account Number) ist     " & vbCr
    A = A & "die internationalisierte Bankkontonummer." & vbCr & vbCr
    A = A & "Die Prüfziffer wird errechnet, indem die IBAN im" & vbCr
    A = A & "erste Schritt nach einem bestimmten Verfahren in" & vbCr
    A = A & "eine Zahl verwandelt wird." & vbCr & vbCr
    A = A & "Diese Zahl wird durch 97 geteilt und der Rest" & vbCr
    A = A & "von 98 subtrahiert." & vbCr & vbCr
    A = A & "Das zweistellige Ergebnis ist die Prüfziffer, die an" & vbCr
    A = A & "der dritten und vierten Stelle der IBAN steht." & vbCr & vbCr
    A = A & "Die IBAN hat maximal 34 Stellen, in Deutschland" & vbCr
    A = A & "hat sie immer 22 Stellen."
    MsgBox A, vbInformation, T
End Sub
```

```
'
' Hilfsfunktionen für Berechnungen usw.
'


Private Function PZ_ISBN(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine ISBN-Nummer;
    ' vorausgesetzt wird, dass Nummer eine neunstellige Zahl enthält
    '
    Dim Zaehler As Integer
    Dim PZ As Integer
    Dim Summe As Long
    Summe = 0
    '
    ' von rechts nach links die 9 Ziffern mit den Gewichten
    ' 2, 3, 4, 5, 6, 7, 8, 9, 10 multiplizieren und Ergebnis summieren
    '
    For Zaehler = 2 To 10
        Summe = Summe + Mid(Nummer, 11 - Zaehler, 1) * Zaehler
    Next Zaehler
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 11
    '
    PZ = 11 - Summe Mod 11
    '
    ' ist die Summe selbst schon eine Elferzahl, so ist die Prüfziffer 0;
    ' wenn der Abstand zur nächsten Elferzahl 10 ist, wird ein X gesetzt
    '
    If PZ = 11 Then
        PZ_ISBN = "0"
    ElseIf PZ = 10 Then
        PZ_ISBN = "X"
    Else
        PZ_ISBN = PZ
    End If
End Function


Private Function PZ_EAN(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine EAN-Nummer;
    ' da die EAN 8 oder 13 Stellen haben kann,
    ' werden 7 oder 12 Stellen erwartet
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' die ungeraden Stellen (von rechts) mit 3 gewichten
    '
    For I = 0 To L - 1 Step 2
        S = S + Mid(Nummer, L - I, 1) * 3
    Next I
    '
    ' die geraden Stellen (von rechts) mit 1 gewichten
    '
    For I = 1 To L - 1 Step 2
        S = S + Mid(Nummer, L - I, 1)
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 10
    '
    PZ = 10 - S Mod 10
    '
```

```
    ' ist die Prüfziffer 10, nur die 0 anhängen
    '
    If PZ = 10 Then
        PZ_EAN = "0"
    Else
        PZ_EAN = PZ
    End If
End Function


Private Function PZ_EKONSm10(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer gemäß den
    ' EKONS-Richtlinien;
    ' die Stellen werden von rechts nach links mit 2 und 1 gewichtet;
    ' der Modulus ist 10
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' die ungeraden Stellen (von rechts) mit 2 gewichten
    '
    For I = 0 To L - 1 Step 2
        S = S + Mid(Nummer, L - I, 1) * 2
    Next I
    '
    ' die geraden Stellen (von rechts) mit 1 gewichten
    '
    For I = 1 To L - 1 Step 2
        S = S + Mid(Nummer, L - I, 1)
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 10
    '
    PZ = 10 - S Mod 10
    '
    ' Ist die Prüfziffer 10, nur die 0 anhängen
    '
    If PZ = 10 Then
        PZ_EKONSm10 = "0"
    Else
        PZ_EKONSm10 = PZ
    End If
End Function


Private Function PZ_EKONSm10Q(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer gemäß den
    ' EKONS-Richtlinien;
    ' die Stellen von rechts nach links mit 2 und 1 gewichten;
    ' aus zweistelligen Produkten die Quersumme bilden;
    ' der Modulus ist 10
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' die ungeraden Stellen (von rechts) mit 2 gewichten
    ' und ggf. die Quersumme ermitteln
    '
    For I = 0 To L - 1 Step 2
```

```
        S = S + Quersumme(Mid(Nummer, L - I, 1) * 2)
    Next I
    '
    ' die geraden Stellen (von rechts) mit 1 gewichten
    '
    For I = 1 To L - 1 Step 2
        S = S + Mid(Nummer, L - I, 1)
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 10
    '
    PZ = 10 - S Mod 10
    '
    ' Ist die Prüfziffer 10, nur die 0 anhängen
    '
    If PZ = 10 Then
        PZ_EKONSm10Q = "0"
    Else
        PZ_EKONSm10Q = PZ
    End If
End Function


Private Function PZ_Mod10G371(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer der
    ' z.B. Württembergischen Sparkassen;
    ' die Stellen von rechts nach links mit 3, 7 und 1 gewichten;
    ' der Modulus ist 10
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' die erste, vierte, siebte usw. Stelle von rechts mit 3 gewichten
    '
    For I = 0 To L - 1 Step 3
        S = S + Mid(Nummer, L - I, 1) * 3
    Next I
    '
    ' die zweite, fünfte, achte usw. Stelle von rechts mit 7 gewichten
    '
    For I = 1 To L - 1 Step 3
        S = S + Mid(Nummer, L - I, 1) * 7
    Next I
    '
    ' die dritte, sechste, neunte usw. Stelle von rechts mit 1 gewichten
    '
    For I = 2 To L - 1 Step 3
        S = S + Mid(Nummer, L - I, 1)
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 10
    '
    PZ = 10 - S Mod 10
    '
    ' Ist die Prüfziffer 10, nur die 0 anhängen
    '
    If PZ = 10 Then
        PZ_Mod10G371 = "0"
    Else
        PZ_Mod10G371 = PZ
    End If
End Function
```

```
Private Function PZ_EKONSm11(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer gemäß den
    ' EKONS-Richtlinien;
    ' die Stellen von rechts nach links mit den Ziffern
    ' 2 bis 10 gewichten; der Modulus ist 11;
    ' Nummern, die eine PZ von 10 ergeben, sind nicht verwendbar
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim G As Integer   ' G = Gewicht bzw. Stellenmultiplikator
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' von rechts nach links die Ziffern mit den Gewichten
    ' 2, 3, 4, 5, 6, 7, 8, 9, 10 multiplizieren und das Ergebnis summieren
    '
    For I = 0 To L - 1
        '
        ' das erste Gewicht ist 2
        ' bei mehr als 9 Stellen beginnen die Gewichte von vorne
        '
        G = I Mod 9 + 2
        '
        ' die nächste Stelle gewichten und das Produkt addieren
        '
        S = S + Mid(Nummer, L - I, 1) * G
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 11
    '
    PZ = 11 - S Mod 11
    '
    ' Ist die Summe selbst schon eine Elferzahl, so ist die Prüfziffer 0;
    ' Wenn der Abstand zur nächsten Elferzahl 10 ist, dann kann die
    ' Nummer nicht vergeben werden
    '
    If PZ = 11 Then
        PZ_EKONSm11 = "0"
    ElseIf PZ = 10 Then
        PZ_EKONSm11 = "Nr. kann nicht vergeben werden!"
    Else
        PZ_EKONSm11 = PZ
    End If
End Function


Private Function PZ_EKONSm11m(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer gemäß den
    ' EKONS-Richtlinien;
    ' die Stellen von rechts nach links mit den Ziffern 2 bis 10 gewichten;
    ' der Modulus ist 11; modifiziertes Verfahren: PZ=10 wird zu PZ=0
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim G As Integer   ' G = Gewicht bzw. Stellenmultiplikator
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' von rechts nach links die Ziffern mit den Gewichten
    ' 2, 3, 4, 5, 6, 7, 8, 9, 10 multiplizieren und das Ergebnis summieren
    '
    For I = 0 To L - 1
```

```
        '
        ' das erste Gewicht ist 2
        ' bei mehr als 9 Stellen beginnen die Gewichte von vorne
        '
        G = I Mod 9 + 2
        '
        ' die nächste Stelle gewichten und das Produkt addieren
        '
        S = S + Mid(Nummer, L - I, 1) * G
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 11
    '
    PZ = 11 - S Mod 11
    '
    ' ist die Summe selbst schon eine Elferzahl, so ist die Prüfziffer 0;
    ' ist die Prüfziffer 10, dann wird sie ebenfalls auf 0 gesetzt
    '
    If PZ = 11 Or PZ = 10 Then
        PZ_EKONSm11m = "0"
    Else
        PZ_EKONSm11m = PZ
    End If
End Function


Private Function PZ_Mod11G27(Nummer As String) As String
    '
    ' Errechnen der Prüfziffer für eine Kontonummer
    ' z.B. der Sparkasse Heidelberg;
    ' die Stellen von rechts nach links mit den Ziffern
    ' 2 bis 7 gewichten; der Modulus ist 11;
    ' modifiziertes Verfahren: PZ=10 wird zu PZ=0
    '
    Dim I As Integer   ' I = Zähler für die Iteration
    Dim L As Integer   ' L = Länge der Nummer
    Dim G As Integer   ' G = Gewicht bzw. Stellenmultiplikator
    Dim S As Long      ' S = Summe der gewichteten Ziffern
    Dim PZ As Integer
    S = 0
    L = Len(Nummer)
    '
    ' von rechts nach links die Ziffern mit den Gewichten 2, 3, 4, 5, 6, 7
    ' multiplizieren und das Ergebnis summieren
    '
    For I = 0 To L - 1
        '
        ' das erste Gewicht ist 2
        ' bei mehr als 6 Stellen beginnen die Gewichte von vorne
        '
        G = I Mod 6 + 2
        '
        ' die nächste Stelle gewichten und das Produkt addieren
        '
        S = S + Mid(Nummer, L - I, 1) * G
    Next I
    '
    ' die Prüfziffer berechnet sich aus dem Abstand der Summe
    ' zum nächst größeren Vielfachen von 11
    '
    PZ = 11 - S Mod 11
    '
    ' ist die Summe selbst schon eine Elferzahl, so ist die Prüfziffer 0;
    ' ist die Prüfziffer 10, ebenfalls auf 0 setzen
    '
    If PZ = 11 Or PZ = 10 Then
        PZ_Mod11G27 = "0"
    Else
        PZ_Mod11G27 = PZ
```

```
    End If
End Function


Private Function PZ_IBAN(IBAN As String) As String
    '
    ' Berechnung der Prüfziffer einer IBAN
    '
    Dim Zahl As String
    Dim PZ As Integer
    '
    ' aus der IBAN eine Zahl machen
    '
    Zahl = IBAN2Z(IBAN)
    '
    ' Zahl durch 97 teilen, Rest ermitteln
    '
    PZ = 98 - Rest97(Zahl)
    '
    ' Ergebnis muss 2stellig sein
    '
    PZ_IBAN = Format(PZ, "00")
End Function


Private Function ClearIBAN(Eingabe As String) As String
    '
    ' Entfernen aller Gliederungs- und Leerzeichen in der IBAN;
    ' Konstante "IBAN" am Anfang entfernen
    '
    Dim IBAN As String
    Dim I As Integer     ' I = Zählvariable
    Dim Z As String      ' Z = (ein) Zeichen
    '
    ' zuerst Leerstellen, Gliederungszeichen usw. entfernen;
    ' Kleinbuchstaben in Großbuchstaben verwandeln
    '
    IBAN = ""
    For I = 1 To Len(Eingabe)
        Z = UCase(Mid(Eingabe, I, 1))
        If (Asc(Z) > 47 And Asc(Z) < 58) Or (Asc(Z) > 64 And Asc(Z) < 91) Then
            IBAN = IBAN & Z
        End If
    Next I
    '
    ' falls am Anfang "IBAN" steht, ebenfalls entfernen
    '
    If Left(IBAN, 4) = "IBAN" Then IBAN = Right(IBAN, Len(IBAN) - 4)
    ClearIBAN = IBAN
End Function


Private Function IsIBAN(Eingabe As String) As Boolean
    '
    ' Prüfen, ob die Eingabe von der Länge her eine gültige IBAN sein kann
    '
    Dim Land As String
    Dim Lang As Integer
    Lang = Len(Eingabe)
    '
    ' Vorbelegung des Ergebnisses;
    ' wird unten keine gültige Kombination gefunden,
    ' dann bleibt der Rückgabewert "Falsch"
    '
    IsIBAN = False
    If Lang > 4 Then
        '
        ' da die IBAN aus den einzelnen Ländern unterschiedliche
        ' Längen haben, Länderkennzeichen berücksichtigen
        '
```

```
        Land = Left(Eingabe, 2)
        Select Case Land
            Case "NO"
                If Lang = 15 Then IsIBAN = True
            Case "BE"
                If Lang = 16 Then IsIBAN = True
            Case "DK", "FI", "NL"
                If Lang = 18 Then IsIBAN = True
            Case "LU", "AT"
                If Lang = 20 Then IsIBAN = True
            Case "CH"
                If Lang = 21 Then IsIBAN = True
            Case "DE", "GB", "IE"
                If Lang = 22 Then IsIBAN = True
            Case "SE", "ES"
                If Lang = 24 Then IsIBAN = True
            Case "PT"
                If Lang = 25 Then IsIBAN = True
            Case "IS"
                If Lang = 26 Then IsIBAN = True
            Case "FR", "IT"
                If Lang = 27 Then IsIBAN = True
            Case "PL"
                If Lang = 28 Then IsIBAN = True
        End Select
    End If
End Function


Function Numerisch(Wert As String) As String
    '
    ' Entferne alle nichtnumerischen Zeichen,
    ' z.B. Gliederungszeichen aus der Nummer
    '
    Dim I As Integer    ' I = Iteration
    Dim Z As String     ' Z = Zeichen
    Numerisch = ""
    For I = 1 To Len(Wert)
        '
        ' Zeichen für Zeichen den Wert untersuchen
        '
        Z = Mid(Wert, I, 1)
        If (Asc(Z) > 47 And Asc(Z) < 58) Then
            '
            ' nur die Ziffern 0 bis 9 übernehmen
            '
            Numerisch = Numerisch + Z
        End If
    Next I
End Function


Private Function Quersumme(Wert As Integer) As Integer
    '
    ' Ermitteln der Quersumme einer Zahl;
    ' so lange 9 abziehen, bis die Zahl kleiner als 10 ist
    ' (in den Prüfziffernverfahren wird von der Quersumme wiederum die
    '  Quersumme verlangt, falls sie zweistellig ist!)
    '
    Quersumme = Wert
    Do While Quersumme > 9
        Quersumme = Quersumme - 9
    Loop
End Function


Private Function IBAN2Z(IBAN As String) As String
    '
    ' IBAN gemäß Vorschrift in eine Zahl umwandeln
    '
```

```
    Dim IBAN2 As String   ' veränderte IBAN, Schritt 1
    Dim Zahl As String    ' veränderte IBAN, nur Ziffern
    Dim I As Integer      ' I = Iteration
    Dim Z As String       ' Z = Zeichen
    '
    ' die ersten 4 Stellen nach rechts verschieben
    '
    IBAN2 = Right(IBAN, Len(IBAN) - 4) & Left(IBAN, 4)
    '
    ' Buchstaben in Zahlen umtauschen
    '
    Zahl = ""
    For I = 1 To Len(IBAN2)
        Z = Mid(IBAN2, I, 1)
        If Asc(Z) > 47 And Asc(Z) < 58 Then
            '
            ' das ist eine Ziffer
            '
            Zahl = Zahl & Z
        ElseIf Asc(Z) > 64 And Asc(Z) < 91 Then
            '
            ' das ist ein Buchstabe
            ' A=10, B=11 usw.
            '
            Zahl = Zahl & Asc(Z) - 55
        End If
    Next I
    IBAN2Z = Zahl
End Function


Private Function Rest97(Zahl As String) As Integer
    '
    ' Divisionsrest von 97 bei langen Zahlen ermitteln
    '
    Dim Teilzahl As Long
    Dim Rest As Integer
    Do While Len(Zahl) > 9
        '
        ' die Division erfolgt in Teilschritten zu 9 Stellen,
        ' da der Datentyp "Long" nur 9stellige Zahlen verkraftet
        '
        Teilzahl = Left(Zahl, 9)
        Rest = Teilzahl Mod 97
        Zahl = Right(Zahl, Len(Zahl) - 9)
        Zahl = Rest & Zahl
    Loop
    Teilzahl = Zahl
    Rest97 = Teilzahl Mod 97
End Function


'
' Fehlerbehandlung
'


Sub Fehler(Fehlerart As Integer, Optional X As Integer)
    '
    ' Fehlermeldung gemäß übergebener Nummer ausgeben
    '
    Dim Fehlertext As String
    Dim Titel As String
    Titel = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    Select Case Fehlerart
        Case 1
            Fehlertext = "Die eingegebene Nummer ist zu lang!" & vbCr & _
                "Es sind nur " & X & " Ziffern erlaubt!"
        Case 2
            Fehlertext = "Die eingegebene Nummer ist zu kurz!" & vbCr & _
```

```
                "Es müssen " & X & " Ziffern sein!"
        Case 3
            Fehlertext = "Die eingegebene Nummer hat nicht die" _
                & " erwartete Länge!"
        Case 4
            Fehlertext = "Die eingegebene Nummer ist zu kurz!" & vbCr & _
                "Es müssen mindestens " & X & " Ziffern sein!"
        Case 5
            Fehlertext = "Die EAN hat ohne Prüfziffer" & vbCr & _
                "7 oder 12 Stellen!"
        Case 6
            Fehlertext = "Die EAN hat 8 oder 13 Stellen!"
        Case 8
            Fehlertext = "Sie haben keine Berechnungsmethode ausgewählt!"
        Case 9
            Fehlertext = "Die Eingabe enthält keine Ziffern!"
        Case 10
            Fehlertext = "Es wurde keine gültige IBAN eingegeben!"
        Case 11
            Fehlertext = "Die Prüfziffer (3. und 4. Stelle)" & vbCr & _
                "nicht mit '00' vorbelegt!"
        Case 12
            Fehlertext = "Die Prüfziffer (3. und 4. Stelle) ist '00'!"
        Case Else
            Fehlertext = "Ungültiger Fehler!"
    End Select
    MsgBox Fehlertext, vbCritical, Titel
End Sub
```

Formular

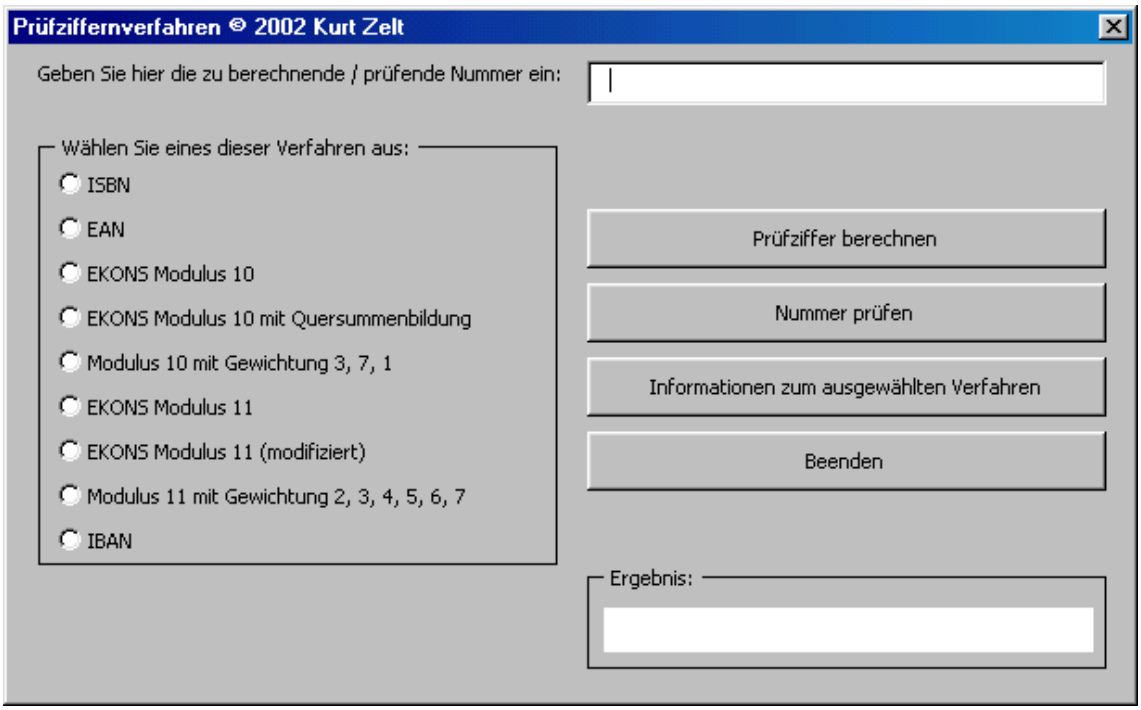

Abbildung 9: Formular frmPZV

Quellcode zum Formular

```
Option Explicit

Private Sub cmdBerechnen_Click()
    Dim Eingabe As String
    Dim Ausgabe As String
    Ausgabe = ""
    Eingabe = txtPZV.Value
    If Eingabe = "" Then
        Call Fehler(9)
    Else
        If optISBN.Value = True Then
            Ausgabe = Errechne_ISBN(Eingabe)
        ElseIf optEAN.Value = True Then
            Ausgabe = Errechne_EAN(Eingabe)
        ElseIf optEKONSm10.Value = True Then
            Ausgabe = Errechne_EKONSm10(Eingabe)
        ElseIf optEKONSm10Q.Value = True Then
            Ausgabe = Errechne_EKONSm10Q(Eingabe)
        ElseIf optMod10G371.Value = True Then
            Ausgabe = Errechne_Mod10G371(Eingabe)
        ElseIf optEKONSm11.Value = True Then
            Ausgabe = Errechne_EKONSm11(Eingabe)
        ElseIf optEKONSm11m.Value = True Then
            Ausgabe = Errechne_EKONSm11m(Eingabe)
        ElseIf optMod11G27.Value = True Then
            Ausgabe = Errechne_Mod11G27(Eingabe)
        ElseIf optIBAN.Value = True Then
            Ausgabe = Errechne_IBAN(Eingabe)
        Else
            '
            ' nichts ausgewählt
            '
            Call Fehler(8)
        End If
    End If
    lblErgebnis.Caption = Ausgabe
    If Left(Ausgabe, 6) = "Fehler" Then
        lblErgebnis.ForeColor = RGB(255, 0, 0)   'rot
    ElseIf Left(Ausgabe, 3) = "Nr." Then
        lblErgebnis.ForeColor = RGB(255, 0, 0)   'rot
    Else
        lblErgebnis.ForeColor = RGB(0, 0, 255)   'blau
    End If
End Sub


Private Sub cmdInfo_Click()
    If optISBN.Value = True Then
        Call Info_ISBN
    ElseIf optEAN.Value = True Then
        Call Info_EAN
    ElseIf optEKONSm10.Value = True Then
        Call Info_EKONSm10
    ElseIf optEKONSm10Q.Value = True Then
        Call Info_EKONSm10Q
    ElseIf optMod10G371.Value = True Then
        Call Info_Mod10G371
    ElseIf optEKONSm11.Value = True Then
        Call Info_EKONSm11
    ElseIf optEKONSm11m.Value = True Then
        Call Info_EKONSm11m
    ElseIf optMod11G27.Value = True Then
        Call Info_Mod11G27
    ElseIf optIBAN.Value = True Then
        Call Info_IBAN
    Else
```

```
        '
        ' nichts ausgewählt
        '
        Call Fehler(8)
    End If
End Sub


Private Sub cmdPrüfen_Click()
    Dim Eingabe As String
    Dim Ausgabe As Boolean
    Eingabe = txtPZV.Value
    If Eingabe = "" Then
        Call Fehler(9)
        Ausgabe = False
    Else
        If optISBN.Value = True Then
            Ausgabe = Prüfe_ISBN(Eingabe)
        ElseIf optEAN.Value = True Then
            Ausgabe = Prüfe_EAN(Eingabe)
        ElseIf optEKONSm10.Value = True Then
            Ausgabe = Prüfe_EKONSm10(Eingabe)
        ElseIf optEKONSm10Q.Value = True Then
            Ausgabe = Prüfe_EKONSm10Q(Eingabe)
        ElseIf optMod10G371.Value = True Then
            Ausgabe = Prüfe_Mod10G371(Eingabe)
        ElseIf optEKONSm11.Value = True Then
            Ausgabe = Prüfe_EKONSm11(Eingabe)
        ElseIf optEKONSm11m.Value = True Then
            Ausgabe = Prüfe_EKONSm11m(Eingabe)
        ElseIf optMod11G27.Value = True Then
            Ausgabe = Prüfe_Mod11G27(Eingabe)
        ElseIf optIBAN.Value = True Then
            Ausgabe = Prüfe_IBAN(Eingabe)
        Else
            '
            ' nichts ausgewählt
            '
            Call Fehler(8)
            Ausgabe = False
        End If
    End If
    If Ausgabe = True Then
        lblErgebnis.Caption = "Nummer korrekt!"
        lblErgebnis.ForeColor = RGB(0, 128, 64)   'grün
    ElseIf Ausgabe = False Then
        lblErgebnis.Caption = "Fehler!"
        lblErgebnis.ForeColor = RGB(255, 0, 0)   'rot
    End If
End Sub


Private Sub cmdEnd_Click()
    '
    ' beim Beenden kann die letzte Nummer in die Excel-Tabelle
    ' übernommen werden
    '
    Dim Nummer As String
    Dim Antwort As Integer
    Nummer = Numerisch(lblErgebnis.Caption)
    '
    ' nur wenn eine Nummer im Ergebnisfeld steht,
    ' den Nutzer fragen, was zu tun ist!
    '
    If Nummer <> "" Then
        Antwort = MsgBox("Soll das Ergebnis in die aktuelle " & _
            "Excel-Zelle geschrieben werden?", vbYesNoCancel, _
            "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt")
        If Antwort = vbYes Then
            '
```

```
            ' Nummer in die Zelle schreiben
            '
            ActiveCell.Value = lblErgebnis.Caption
        ElseIf Antwort = vbCancel Then
            '
            ' Wenn Cancel gedrückt wird, zurück zum Formular
            '
            Exit Sub
        End If
    End If
    '
    ' Formular erst ausblenden und dann schliessen (wirkt so schneller)
    '
    Me.Hide
    Unload Me
End Sub


Private Sub txtPZV_Change()
    '
    ' wenn das Eingabefeld verändert wird,
    ' das Ausgabefeld zurücksetzen!
    '
    lblErgebnis.Caption = ""
    lblErgebnis.ForeColor = RGB(0, 0, 0)   'schwarz
End Sub


Private Sub UserForm_Initialize()
    Me.Caption = "Prüfziffernverfahren " & Chr(169) & " 2002 Kurt Zelt"
    '
    ' Vorbelegen des Eingabefeldes mit dem Wert, der in der
    ' aktiven Excel-Zelle vorliegt
    '
    txtPZV.Value = ActiveCell.Value
End Sub
```

Quellenverzeichnis

Literatur

Bernhardt, Rolf und Bernhardt, Werner, Nummerungssysteme: Grundbegriffe und Einführung, Systemvergleiche, prakt. Anwendung, speziell bei EDV-Systemen, 2. Aufl., Ehningen bei Böblingen 1990

Fuhrberg, Kai; Häger, Dirk und Wolf, Stefan, Internet-Sicherheit: Browser, Firewalls und Verschlüsselung, 3. Aufl., München und Wien 2001

Grupp, Bruno, Optimale Verschlüsselung bei Online-Datenverarbeitung: Aufbau moderner Nummernsysteme jeder Art, Personennummern und Auftragsnummern, 1. Aufl., Köln 1987

Hansen, Hans Robert, Wirtschaftsinformatik I, Grundlagen betrieblicher Informationsverarbeitung, 7. Aufl., Stuttgart 1998

Jahnke, Bernd, Gestaltung leistungsfähiger Nummernsysteme für die DV-Organisation, 1. Aufl., München 1979

Kunerth, Walter und Werner, Günter, EDV-gerechte Verschlüsselung: Grundlagen und Anwendung moderner Nummernsysteme, 2. Aufl., Stuttgart 1981

Schiebel, Walter, Die Europäische Artikelnummer (EAN): Ausbreitung und Anwendung in Warenwirtschaftssystemen (mit den offiziellen EAN-Richtlinien), 1. Aufl., Wien 1987

Schneider, Wolfgang und Lauterbach, Heinz, Betriebliche Nummernsysteme für die elektronische Datenverarbeitung: Wie verschlüsselt man verarbeitungsgerecht die wesentlichen Betriebsdaten in Wirtschaft und Verwaltung?, 1. Aufl., Heidelberg 1971

Schwarzenbach, Fritz H., Numerische Codierung: die zahlenmäßige Verschlüsselung von Aussagen als Arbeitsinstrument der manuellen und elektronischen Datenverarbeitung, Verfahren und Anwendungen, 1. Aufl., Bern und Stuttgart 1977

Spona, Helma, Visual Basic 6, 1. Aufl., Kaarst 2000

Stahlknecht, Peter und Hasenkamp, Ulrich, Einführung in die Wirtschaftsinformatik, 10. Aufl., Berlin und Heidelberg 2002

o.V., DIN 6763: Nummerung: Allgemeine Begriffe, Hrsg. Deutscher Normenausschuss, Berlin und Köln 1972

o.V., Einheitliches Kontonummernsystem der Sparkassen und Girozentralen (EKONS), Hrsg. Deutscher Sparkassen- und Giroverband e.V. Bonn, 1. Aufl., Stuttgart 1967

o.V., Grundlagen des Giroverkehrs, Hrsg. Deutscher Sparkassen- und Giroverband, Berlin, Fassung III/2002, Berlin 2002

o.V., IBAN: International Bank Account Number, Hrsg. European Committee for Banking Standards, 1. Aufl., Brüssel 1996

o.V., Register of European Account Numbers, Hrsg. European Committee for Banking Standards, Aufl. Mai 2002, Brüssel 2002

Internet

Kotthaus, Michael, Prüfziffernberechnung, Stand: Juli 2002, http://wrstud.urz.uni-wuppertal.de/~ea0449/pruefz.htm

o.V., Prüfzifferberechnungsmethoden zur Prüfung von Kontonummern auf ihre Richtigkeit, Hrsg. Deutsche Bundesbank Z 11-5, Stand: Juni 2002, http://www.bundesbank.de/zv/zv_pruefziffern.php

o.V., Bankleitzahlen-Datei, Hrsg. Deutsche Bundesbank, Stand: Juni 2002, http://www.bundesbank.de/zv/zv_bankleitzahlen.htm

o.V., IBAN, Hrsg. Pruefziffernberechnung.DE, Stand: Juni 2002, http://www.pruefziffernberechnung.de/I/IBAN.shtml

o.V., Prüfzifferberechnung, IBAN * IPI, Der neue Zahlungsbeleg, Hrsg. Swiss Interbank Clearing AG, Zürich 2000, http://www.telekurs-sic.ch

Auskünfte

Auskunft per E-Mail durch Herrn Wolfgang Hintze, Pressestelle der Deutschen Bundesbank vom 22.07.2002

Abbildungsverzeichnis

Abbildung 1: Aufbau der IBAN der europäischen Länder 36

Abbildung 2: Formular frmPZV mit Rechenergebnis 45

Abbildung 3: Formular frmPZV mit Prüfergebnis 45

Abbildung 4: Infofenster 46

Abbildung 5: Frage beim Beenden 46

Abbildung 6: Start über die Symbolleiste 47

Abbildung 7: Nummer korrekt oder nicht 47

Abbildung 8: Beispiel einer Fehlermeldung 49

Abbildung 9: Formular frmPZV 77

Eidesstattliche Versicherung

Hiermit versichere ich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt worden ist, insbesondere, dass ich alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen sind, durch Zitate als solche kenntlich gemacht habe.

Mannheim, den 7. September 2002