Access 2021

Automatisierung, Programmierung

Ricardo Hernández García

1. Ausgabe, Januar 2022

ISBN 978-3-98569-060-2

ACC2021P

Plus

HERDT

1

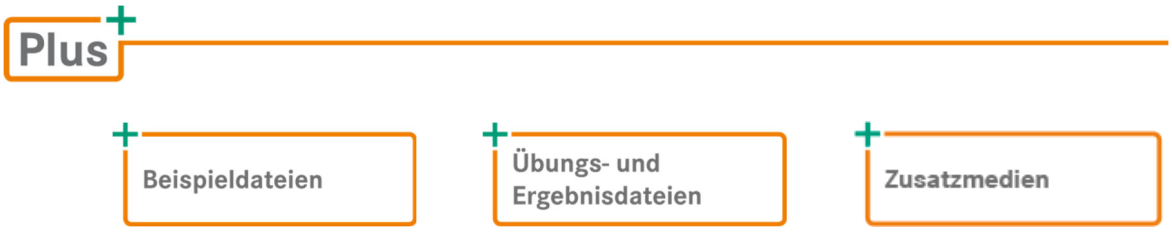
Bevor Sie beginnen …

HERDT BuchPlus – unser Konzept:

Problemlos einsteigen – Effizient lernen – Zielgerichtet nachschlagen

(weitere Infos unter *www.herdt.com/BuchPlus*)

Nutzen Sie unsere maßgeschneiderten, im Internet frei verfügbaren Medien:



Wie Sie schnell auf diese BuchPlus-Medien zugreifen können, erfahren Sie unter *www.herdt.com/BuchPlus*

Um die Lerninhalte des Buches praktisch nachzuvollziehen, benötigen Sie:

- Windows 10
- Access 2021

Beachten Sie, dass es bei einem Teil der Beispiele und Übungen zu Problemen mit Zugriffsrechten kommen kann, wenn sie von einem Netzlaufwerk aus gestartet werden. Es ist daher zu empfehlen, die Beispiele und Übungen auf ein Laufwerk des Rechners zu kopieren und sie von dort aus zu starten.

Empfohlene Vorkenntnisse

Um sich problemlos die Grundlagen der Programmierung mit Access 2021 aneignen zu können, sollten Sie bereits über folgende Kenntnisse und Fähigkeiten verfügen:

- Datenbank erstellen und verwalten sowie Access 2021 konfigurieren
- Tabellen entwerfen und mit Tabellendaten arbeiten
- Primärschlüssel, Primärindex sowie Sekundärindizes kennen und einsetzen
- Formulare und Berichte entwerfen und bearbeiten
- Abfragen entwerfen und ausführen
- Grundlagen der Datenbankabfragesprache SQL

Verfügen Sie über Kenntnisse in der Programmiersprache Visual Basic (VB), können Sie diese in Access 2021 nutzen. Visual Basic und Visual Basic for Applications (VBA) gehören zur gleichen Technologiefamilie. Der Unterschied zwischen beiden besteht darin, dass …

- **VB** eine Programmiersprache zur Entwicklung eigenständiger Windows-Anwendungen ist,
- **VBA** dagegen ausschließlich der Automatisierung und Anpassung der Anwendungsprogramme dient, in die es eingebettet ist.

Hinweise zu Soft- und Hardware

In den Funktionsbeschreibungen des Buches wird von einer Erstinstallation von Office 2021 unter dem Betriebssystem Windows 10 und den Standardeinstellungen für die Office-Anwendungen ausgegangen. Abhängig von der Bildschirmauflösung bzw. der Hardware Ihres Computers kann das Aussehen der Symbole und Schaltflächen in Access 2021 und die Fensterdarstellung unter Windows 10 gegebenenfalls von den Abbildungen im Buch abweichen.

Typografische Konventionen

Damit Sie bestimmte Elemente auf einen Blick erkennen und zuordnen können, werden diese im Text durch eine besondere Formatierung hervorgehoben. So werden beispielsweise Bezeichnungen für Programmelemente wie Register oder Schaltflächen immer *kursiv* geschrieben und wichtige Begriffe **fett** hervorgehoben.

`Courier New`	kennzeichnet Programmtext, Programmnamen, Funktionsnamen, Variablennamen, Datentypen, Operatoren etc.
`Courier New Kursiv`	kennzeichnet Zeichenfolgen, die vom Programm ausgegeben oder ins Programm eingegeben werden.

2

Access programmieren

2.1 Ausgangslage der Access-Programmierung

Entwicklung neuer Anwendungen

In den einzelnen Programmen aus der Office-Reihe von Microsoft wie z. B. Access, Excel, PowerPoint oder Word ist die Möglichkeit implementiert, durch eigene Programmierung den Funktionsumfang dieser Programme benutzerspezifisch zu erweitern.

An manchen Stellen geht es dabei nur darum, bestimmte immer in gleicher oder ähnlicher Weise wiederkehrende Aufgaben in Form von kleinen Unterprogrammen zu automatisieren und dann per „Knopfdruck“ (meistens über das Anklicken einer Schaltfläche) ausführen zu lassen.

Andere Programmiervorhaben gehen aber weit darüber hinaus und enden darin, dass komplett neue Anwendungen zu den unterschiedlichsten Zwecken entstehen und die Office-Software dann lediglich die Basis für diese Applikationen stellt.

In vielen dieser Projekte taucht fast zwangsläufig irgendwann die Situation auf, dass Daten dauerhaft gespeichert werden müssen. Dafür ist z. B. Excel mit seinen Tabellen in den Arbeitsmappen gut geeignet, aber die ideale Software aus der Office-Reihe, um Daten sicher und effizient langfristig zu speichern, ist Access. Aus diesem Grund wird dieses Programm sehr oft für die Entwicklung neuer Anwendungen eingesetzt.

Access als Basis

Access ist ein Datenbankmanagementsystem oder, anders gesagt, eine Software, die in der Lage ist, Datenbanken zu verwalten. Zwar ist Access nicht vergleichbar mit Programmen wie Microsoft SQL Server oder Oracle, die als hochleistungsfähige Datenbankserver konzipiert sind, aber die Grundmechanismen für die sichere und kontrollierte Aufbewahrung und Überwachung von Daten sind auch in Access vollständig verfügbar.

Fast alle weltweit eingesetzten Datenbanken sind relationale Datenbanken. Access stellt alle grundlegenden Konzepte einer relationalen Datenbank zur Verfügung.

Besonderheiten bei Access

Im Gegensatz zu den Anwendungsprogrammen Excel und Word gibt es bei Access keine Rekorderfunktion. Es ist also nicht möglich, die Durchführung mehrerer Arbeitsschritte nacheinander aufzeichnen zu lassen und danach den entstandenen VBA-Quellcode weiter zu bearbeiten.

Dies ist allerdings kein Nachteil, sondern entspricht der Tatsache, dass bei Access weniger das Automatisieren von Vorgängen im Vordergrund steht, sondern eben mehr das Entwickeln komplett neuer Applikationen.

2.2 Programmiermöglichkeiten

VBA in Access

Access stellt neben Makros eine weitere Möglichkeit zur Automatisierung und Programmierung zur Verfügung: Visual Basic for Applications (kurz VBA). Mit VBA können Sie Ihre Datenbank individuell gestalten und speziellen Bedürfnissen anpassen.

VBA ist die gemeinsame Programmiersprache der Anwendungsprogramme von Microsoft Office, wie Word, Excel und Access. Die grundlegenden Sprachelemente von VBA sind in diesen Anwendungen gleich. Sie unterscheiden sich nur in den Funktionen, die für die jeweilige Anwendung spezifisch sind (beispielsweise die Arbeit mit Datenbanktabellen und Abfragen in Access). Es ist daher besonders einfach, auf Daten in Access-Datenbanken zuzugreifen, diese zu verändern und zu speichern. VBA enthält jedoch auch viele Leistungsmerkmale einer allgemeinen Programmiersprache, wie beispielsweise Visual Basic oder Delphi.

Um die Programmierung zu erleichtern, verfügt Microsoft Office über eine VBA-Entwicklungsumgebung, die in einem eigenen Fenster ausgeführt wird. In der übersichtlichen Entwicklungsumgebung stehen Ihnen viele praktische Programmierhilfen und Arbeitserleichterungen zur Verfügung.

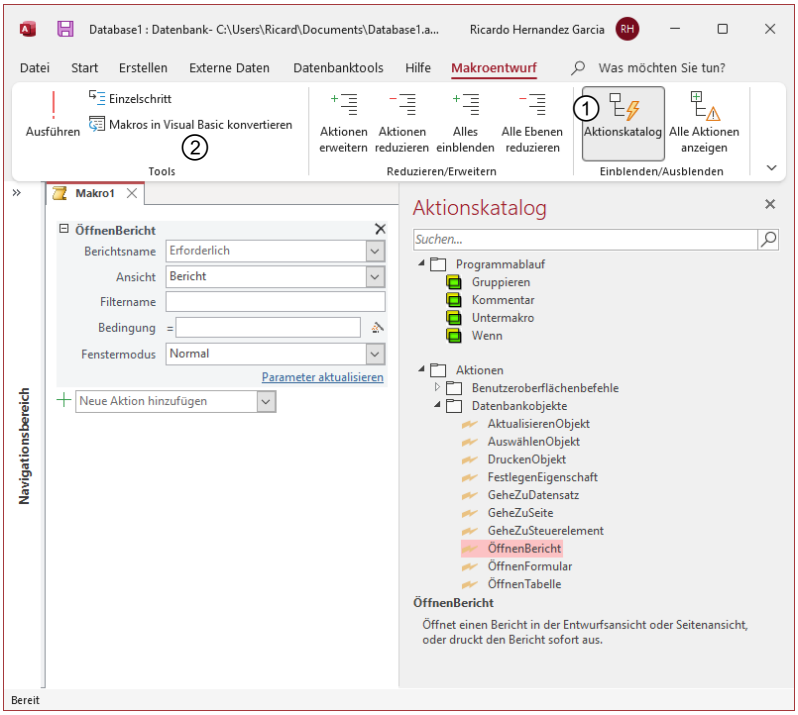
Makros in Access

Der Begriff **Makro** wird bei Access anders verwendet als bei Word oder Excel und bezieht sich hier nicht auf Programme, die mit der Sprache VBA erstellt worden sind.

Ein Makro besteht aus einer Folge von Befehlen. Diese Befehle werden einer Liste von deutschsprachigen Befehlen entnommen, sogenannten Makroaktionen, die – in einem Aktionskatalog ① (Abb. Folgeseite) kategorisiert – eine Vielzahl von Standardsituationen abdecken, die in fast jedem Programm vorkommen können.

Da die meisten der hier aufgelisteten Befehle selbsterklärend sind und der Makro-Generator viel Unterstützung in Bezug auf den Einsatz der Argumente der Befehle bietet, ist hier Programmierung ohne viele Vorkenntnisse möglich.

Access 2021 bietet die Möglichkeit, hier erstellte Makros in Visual Basic zu konvertieren ②. Dies sollten Sie nutzen, wenn Sie auf Basis vorhandener Standardaktionen weitere komplexere Funktionalität implementieren möchten. Sie können dann ohne Programmieraufwand den generierten fehlerfreien Programmcode weiterverwenden und auf diesem aufbauen.



Makroerstellung mithilfe des Aktionskatalogs

Sondermakro „AutoExec“

Beim Öffnen einer Datenbank sucht Access nach einem Makro mit dem Namen „AutoExec“. Falls ein Makro mit diesem Namen vorhanden ist, wird es automatisch gestartet und alle Aktionen, die in diesem Makro hinterlegt sind, werden der Reihe nach abgearbeitet.

Häufig sind hier Befehle zu finden, mit denen z. B. ein bestimmtes Formular geöffnet und dann auf ganze Bildschirmgröße maximiert wird.

2.3 Unterschiede zwischen VBA und Makros

Einsatzgebiete von Makros

Viele Arbeitsschritte können Sie in Access mithilfe von Makros einfach automatisieren. Damit Sie nicht alle Befehle einzeln über Menüpunkte durchführen müssen, können Sie die Aktionen in einem Makro zusammenfassen. Das Makro kann anschließend beliebig oft aufgerufen werden und führt die enthaltenen Aktionen automatisch aus. Mit Makros lösen Sie einfache Problemstellungen und arbeiten Routineaufgaben ab. So lassen sich im Wesentlichen die Arbeiten automatisieren, die über Menübefehle und Symbole durchgeführt werden können. Für die Erstellung von Makros sind keine Kenntnisse einer Programmiersprache erforderlich.

Einer der Hauptvorteile von Makros ist, dass sie von jedem anderen Objekt der Datenbank aus direkt erreichbar sind. Als Beispiel sei ein Makro erwähnt, das das aktuelle Fenster schließt. Dieses eine Makro kann in Hunderten von Formularen zu dem erwähnten Zweck (Formularfenster schließen) eingesetzt werden, muss aber nur ein einziges Mal erstellt und zur Verfügung gestellt werden.

Einsatzgebiete von VBA

Mit VBA besitzen Sie umfangreichere Möglichkeiten, um die Arbeit von Access zu beeinflussen. Hier ist es möglich, auch sehr komplexe Aufgaben abzuarbeiten, beispielsweise komplexe Berechnungen mit Daten aus der Datenbank durchzuführen. VBA erlaubt Ihnen außerdem, die Datenbankanwendung durch den automatisierten Einsatz von Formularen und Steuerelementen benutzerfreundlicher zu gestalten.

Die folgende Auswahl zeigt Ihnen einige Aufgaben, die Sie mit VBA realisieren können:

- benutzerdefinierte Befehlsleisten mit umfangreichen Funktionen,
- benutzerdefinierte Dialogfenster zur Abfrage von Eingabewerten oder zur Auswahl von Optionen,
- Programmsteuerung über Steuerelemente,
- Abfangen von fehlerhaften Eingaben und Anzeigen individueller Fehlermeldungen,
- Durchführen von komplexen Berechnungen und Datenmanipulationen,
- anwenderfreundliches Arbeiten über Formulare,
- Integration von anderen Office-Anwendungen.

Unterschiede zwischen Makros und VBA

Während Makros für einfachere Aufgaben von wenigen Aktionszeilen geeignet sind, lassen sich mit VBA beinahe beliebig komplexe Programme für umfangreiche Datenbankanwendungen entwickeln.

Ob für die Bewältigung einer gegebenen Aufgabe besser ein Makro oder eine VBA-Prozedur verwendet wird, ist neben der vorgesehenen Bereitstellung oder Verteilung der Datenbank auch eine Frage der persönlichen Vorlieben des Entwicklers. Die folgende Tabelle kann bei der Wahl des für einen bestimmten Zweck geeigneten Mittels helfen.

Makros	VBA
Keine Programmierkenntnisse notwendig	Programmierkenntnisse notwendig
Schnelles Lösen einfacher Aufgaben	Umfangreiche Programmiersprache
Begrenzter Befehlsumfang	Viele flexible Befehle und Funktionen
Kaum anpassungsfähig	Strukturierungsmöglichkeiten
Deutsche Befehle	Englische Befehle
Relativ langsame Ausführung	Relativ schnelle Ausführung
In der gesamten Datenbank verfügbar	Verfügbarkeit davon abhängig, wo der Quellcode gespeichert ist
Auch in Access-Webdatenbanken einsetzbar	Nicht in Access-Webdatenbanken einsetzbar

3

Einführung in die Programmierung

Beispieldatei: *Gehalt.accdb*

3.1 Grundlagen der Programmentwicklung

Algorithmen

Damit ein Computer verschiedene Arbeitsschritte automatisch erledigen kann, müssen diese vorher genau festgelegt werden. Ein Algorithmus ist eine Folge von Anweisungen, die solche Arbeitsschritte detailliert beschreiben. Jedes Problem, dessen Lösung durch einen Algorithmus beschrieben werden kann, wie beispielsweise eine Adressenverwaltung, das Auswerten von Statistiken oder automatische Berechnungen, ist im Prinzip durch einen Computer lösbar.

Mit Programmiersprachen wird das Kommunikationsproblem zwischen Mensch und Maschine gelöst. Eine Programmiersprache dient zur Formulierung von Algorithmen auf Computeranlagen. Sie besteht aus einer Reihe von Befehlen oder Anweisungen, deren Funktionen je nach Programmiersprache für bestimmte Anwendungsgebiete geeignet sind.

Ein Programm besteht aus einem oder mehreren Algorithmen, die als Anweisungen einer Programmiersprache (beispielsweise C++ , C# oder Visual Basic) formuliert sind. In einem Programm stehen somit Anweisungen, die der Computer versteht und umsetzen kann.

Interpreter und Compiler

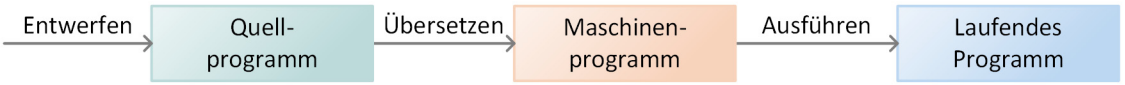
Der Zentralprozessor eines Computers kann nur Maschinenbefehle lesen. Alle Programme müssen erst in Maschinensprache übersetzt werden, bevor sie verarbeitet werden können. Diese Aufgabe übernehmen Übersetzungsprogramme, die als **Interpreter** beziehungsweise **Compiler** bezeichnet werden.

Interpreter (Simultandolmetscher)	Interpreter sind Übersetzungsprogramme, die Anweisung für Anweisung eines Programms übersetzen und ausführen. Dabei wird jeweils nur die gerade bearbeitete Anweisung übersetzt. Der Vorteil besteht in der schnellen Erstellung von Programmen und dem einfachen Testen. Programme können während des Ablaufes unterbrochen, Befehle können rasch geändert, ersetzt oder weggelassen werden.

Compiler (komplette Übersetzung)	Compiler übersetzen Quellprogramme vollständig in Maschinensprache. Dadurch entstehen sogenannte ausführbare Dateien, die als eigenständige Programme gestartet werden können. Mit einem Compiler übersetzte Programme zeichnen sich durch eine höhere Ablaufgeschwindigkeit aus.

VBA verwendet eine Zwischenlösung: Der eingegebene Programmcode wird in einen Zwischencode kompiliert, der während der Ausführung in Maschinencode übersetzt wird. Dieser Kompromiss bietet Vorteile beider Verfahren, da der Programmablauf während der Ausführung überwacht werden kann wie bei einem reinen Interpreter, während zugleich durch den schneller übersetzbaren Zwischencode die Verarbeitungsgeschwindigkeit verbessert wird.

Vorgang der Programmentwicklung



- Zuerst wird ein Quellprogramm entworfen, das den Lösungsweg eines Problems in Form eines Algorithmus enthält. Das Quellprogramm befindet sich im Arbeitsspeicher des Computers.
- Mithilfe eines Übersetzungsprogramms wird das Quellprogramm in Maschinensprache übersetzt; interpretierte Programme werden dabei sofort ausgeführt. Bei fehlerhaften Eingaben wird die Übersetzung abgebrochen.
- Wenn das Quellprogramm erfolgreich übersetzt wurde, steht ein lauffähiges Programm zur Verfügung. Nun muss geprüft werden, ob das Programm den angestrebten Zweck korrekt erreicht; gegebenenfalls muss weiter korrigiert werden.

In der Praxis werden Programme zumeist in kleinen Schritten entwickelt, bei denen kleine Bestandteile des Programms einzeln getestet werden.

Vielfalt der Programmiersprachen

Mittlerweile existieren mehrere hundert Programmiersprachen, die sich nach ihrer historischen Entwicklung und nach ihren Anwendungsgebieten klassifizieren lassen. Dieses Buch beschränkt sich auf die Behandlung von VBA.

3.2 Strukturierte Programmierung

Wichtige Ziele der strukturierten Programmierung

Bereits kleinere Programmierprojekte weisen häufig eine hohe Komplexität auf, die eine strukturierte Vorgehensweise erfordert, um sie in eine Programmiersprache umzusetzen. Das Prinzip der strukturierten Programmierung beruht allgemein auf dem Gedanken, eine Aufgabe durch Aufteilung in Teilaufgaben zu entwickeln und zu lösen. Hierzu gibt es verschiedene Strukturierungsmethoden und Vorgehensweisen.

Vorteile der strukturierten Programmierung

- Logische Fehler des Programmablaufs werden durch strukturierte Programmierung schneller erkannt.
- Änderungen an einem Programm, die zu einem späteren Zeitpunkt notwendig werden, lassen sich effektiver und schneller durchführen.
- Der Aufwand eines Programmierprojektes wird verringert.
- Die Struktur des Programms lässt sich leicht nachvollziehen, da übersichtliche und verständliche Programme entwickelt werden.

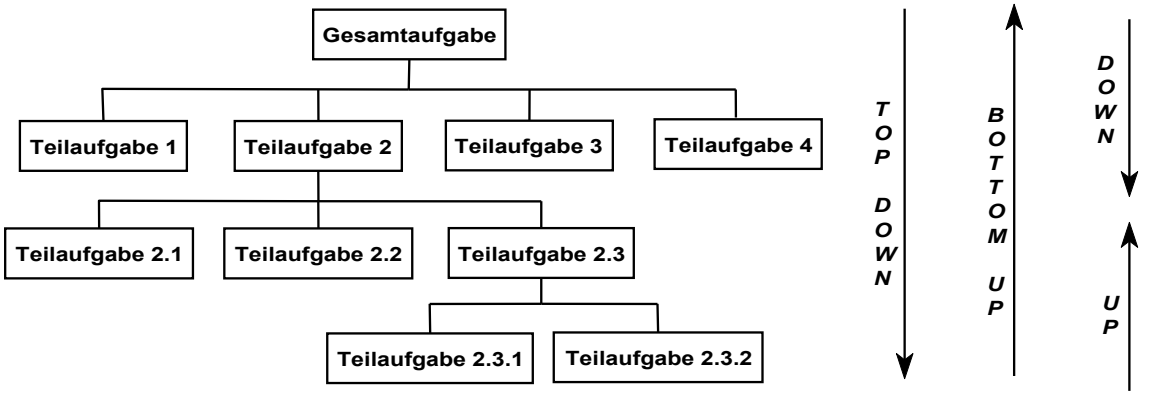
Das Prinzip der schrittweisen Verfeinerung

Um eine Aufgabe in eine Programmiersprache umzusetzen, empfiehlt es sich, die Gesamtaufgabe in verschiedene Teilaufgaben zu zerlegen.

Die Teilaufgaben können Sie wiederum in kleinere Teilaufgaben zerlegen. Diese Strukturierungsmethode wird auch als **schrittweise Verfeinerung** oder **Top-down-Methode** bezeichnet, da eine allgemeine Aufgabe von oben nach unten schrittweise in spezielle Teilaufgaben zerlegt wird.

Weitere Strukturierungsmethoden

Bottom-up-Methode	Hier werden Einzelaufgaben zu einer Gesamtaufgabe zusammengefasst. In der Praxis wird diese Methode meistens nur in Verbindung mit der Top-down-Methode eingesetzt, damit der Bezug zur Gesamtaufgabe nicht verloren geht.
Up-down-Methode (Gegenstromverfahren)	Bei dieser Strukturierungsmethode wird die Gesamtaufgabe top-down verfeinert und Teilaufgaben werden bottom-up verallgemeinert. Auf diese Weise können kritische Teilaufgaben zuerst getestet werden.



Zusammenfassendes Modell der Strukturierungsmethoden

Vorgehensweise der strukturierten Programmierung

Die Lösung eines Problems lässt sich in typische Arbeitsschritte einteilen, die beim Entwurf eines Programms zu beobachten sind. Dieses Schema wird auch als **Software-Lebenszyklus** bezeichnet.

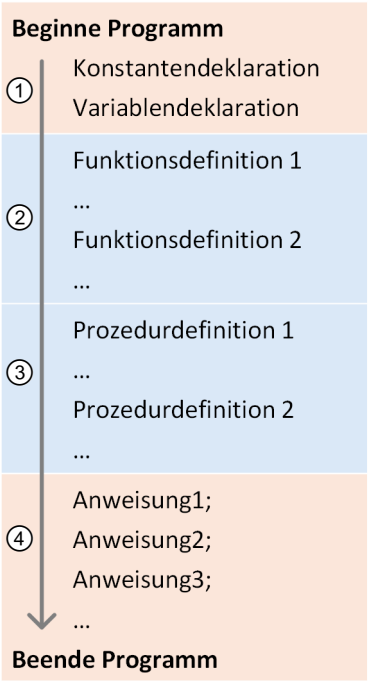
Arbeitsschritte	Aufgaben
1. Analyse der Problemstellung	Zu Beginn muss die Aufgabenstellung genau präzisiert werden. Oft treten noch Unklarheiten auf, beispielsweise zwischen dem Auftraggeber und dem Programmierer. Zuerst müssen alle notwendigen Informationen gesammelt werden, die für die Lösung des Problems wichtig sind. Daraus ergibt sich eine grobe Gliederung der Gesamtaufgabe, die nach dem Prinzip der schrittweisen Verfeinerung in Teilaufgaben zerlegt wird. Für jede Teilaufgabe werden die Arbeitsschritte 1–3 durchgeführt.
2. Inhaltlicher Entwurf	Wenn die Problemstellung in allen Einzelheiten klar ist, können die wichtigsten Elemente benannt werden, die für das Programm notwendig sind, beispielsweise Variablen- und Konstantennamen, Dateinamen, Funktionen, Beziehungen zwischen Daten sowie die Art und Weise, wie Benutzer mit dem Programm kommunizieren sollen (Ein- und Ausgabevariablen).
3. Entwurf der Programmstruktur	Nachdem der inhaltliche Entwurf feststeht, kann der Programmablauf mit grafischen Hilfsmitteln (Programmablaufpläne, Struktogramme) oder durch einen Pseudocode dargestellt werden. Hierbei werden die benötigten Kontrollstrukturen (Folge, Auswahl, Wiederholung) festgelegt.
4. Erstellen des Quellprogramms	Mithilfe des Entwurfs kann der Programmablauf in eine Programmiersprache, wie beispielsweise Visual Basic oder VBA, übertragen werden.
5. Testen	Nachdem das Quellprogramm erstellt wurde, wird es an verschiedenen Beispielen getestet, um die Fehlerfreiheit des Programms zu prüfen.
6. Installation	Nach einem ausführlichen Test des Programms kann es am Arbeitsplatz installiert werden.
7. Dokumentation und Pflege	In der Praxis müssen Programme oft veränderten Situationen angepasst werden. Damit der Programmierer auch zu einem späteren Zeitpunkt die Funktionsweise des Programms nachvollziehen kann, ist es wichtig, alle Informationen (inhaltlicher Entwurf, Programmablaufplan, Struktogramm, Pseudocode) zu Dokumentationszwecken aufzubewahren.

3.3 Modular und prozedural programmieren

Programmiertechnik

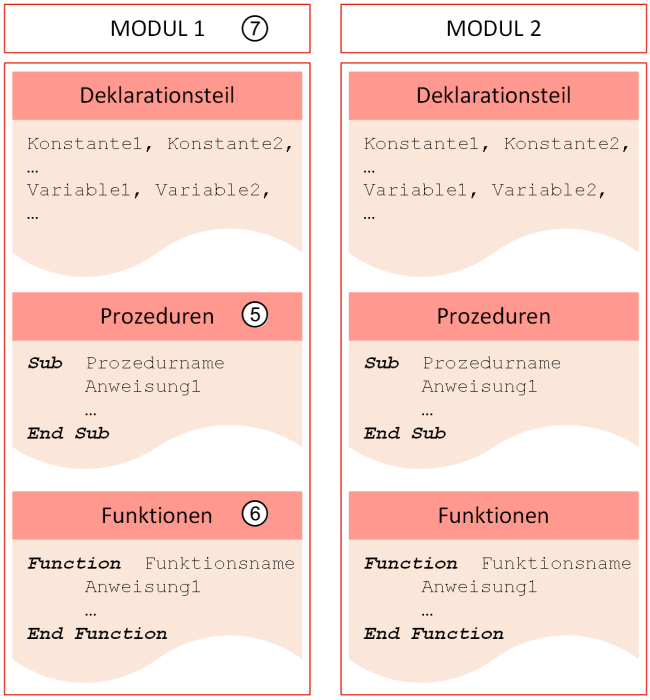
Ein Programm besteht aus Anweisungen und Programmteilen, die in einer vorgegebenen Reihenfolge angeordnet sind und in einer bestimmten Folge abgearbeitet werden. Dabei gibt es zwei grundsätzlich verschiedene Arten des Programmablaufs, die für das Verständnis eines Programms wichtig sind.

Prozedurales Programmschema



Prozedurales Programmschema

Modulares Programmschema



Modulares Programmschema

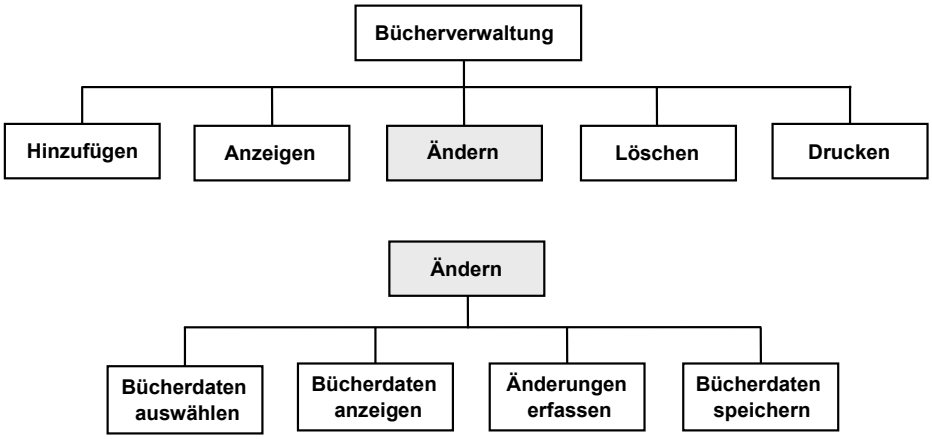
Beim prozeduralen Programmschema stehen die Anweisungen hintereinander. Die Programmteile sind mitunter fest vorgeschrieben, so zum Beispiel die Programmteile ① und ④. Einzelne wiederkehrende Aufgaben lassen sich in Prozeduren ③ oder Funktionen ② kapseln, die im gesamten Programm verfügbar sind. Der Programmablauf ist linear, das heißt, alle Anweisungen werden mit der ersten Anweisung beginnend von oben nach unten abgearbeitet. Beispiele für diese Art der Programmierung sind Pascal, C oder Cobol.

Das modulare Programmschema wird dagegen von ereignis- oder objektorientierten Programmiersprachen (z. B. Visual Basic oder Delphi) verwendet. Solche Programme bestehen aus einer Zusammenstellung von Prozeduren ⑤ bzw. Funktionen ⑥, in sogenannten Modulen ⑦. Jedes Modul ist ein Behälter für Prozeduren, die gemeinsam eine Aufgabe erfüllen. Die in Modulen enthaltenen Prozeduren können bei bestimmten Ereignissen aufgerufen werden, wie zum Beispiel dem Anklicken einer Schaltfläche.

3.4 Mit Modulen strukturiert programmieren

Beispiel

Ein Beispiel für die Anwendung der strukturierten Programmierung ist eine Datenbank zur Benutzerverwaltung einer Telefongesellschaft.



Strukturierung einer Aufgabe am Beispiel einer Benutzerverwaltung

Der Zweck der Datenbank ist das Verwalten von Benutzerdaten. Eine Teilaufgabe ist zum Beispiel das Ändern von Benutzerdaten, die bereits gespeichert wurden. Um diese Änderungen durchzuführen, sind wiederum einzelne Aufgaben notwendig. So müssen die Daten der Benutzer angezeigt und Änderungen erfasst werden.

Bei der strukturierten Programmierung in Access gehen Sie wie folgt vor:

- Gliedern Sie Ihre Aufgabenstellung in logische Teilaufgaben.
- Legen Sie für jede Teilaufgabe ein Modul an.
- In den Modulen definieren Sie die für die Teilaufgabe notwendigen Prozeduren.
- Bei späteren Veränderungen bearbeiten Sie nur das entsprechende Modul.

Denken Sie daran, die Gesamtaufgabe beim strukturierten Programmieren nicht in zu viele Teilaufgaben zu zerlegen. Sonst kann sich der Vorteil der besseren Übersichtlichkeit schnell ins Gegenteil umkehren.