福山平成大学経営学部紀要
第 14 号（2018），29－43 頁

ファイル処理とデータベースに関する学習支援用モデル教材の開発

佐藤 真司

福山平成大学経営学部経営学科

要旨：パーソナルコンピューターを使用した学習支援の仕組みは、学校教育、企業研修等、様々な場面で利用されている。近年では、若年層に対するプログラミング教育を普及するための取り組みや、学習教材、e ラーニング教材、支援ツール等が多数考案されている。本稿では、主としてファイル処理及びデータベースの知識や技能の習得を効果的に行う目的で､C 言語を利用した学習支援用のモデル教材を開発した。また、これらの評価とプログラムの概要について解説を加える。

キーワード：ファイル処理、データベース、学習支援、モデル教材、プログラミング、C 言語

1. はじめに

福山平成大学経営学部経営学科ではコース制を導入している。その一つに経営情報コースを設置しており、ICT（情報通信技術）活用能力の習得、資格取得等を意識した指導を行っている。経営情報コースでは、C 言語、プログラム設計、データベース等の授業を開講している。これらの授業では、データ構造、アルゴリズム、プログラミング、データベース等の分野を扱うことにより、システム設計の基礎知識や応用技術の習得を目指しており、情報システムの開発全般について幅広く指導を行っている。

プログラミングの授業では C 言語を用いた実習を取り入れている。授業は C 言語の基本的な文法を学ぶことから始め、簡単なプログラムを作成することで理解を深めている。しかし、経営学科の学生にはプログラミングの初学者が多く、複雑かつ応用的なプログラムの作成にまで至る者は少ない。特に、ファイル処理はプログラムを作成する上で重要な手法の一つであるが、一般的な入門書では後半の章で扱われることが多く、内容も少し難しいことから、理解が不十分なためにプログラムの作成に手間取る者も見受けられる。

また、入門書等で扱われるファイル処理は、その目的である文法の習得を主体としていることから、ファイル処理に関する様々な操作を紹介しているものは少ない。会社の業務で活用可能なアルゴリズムや応用例を紹介した書籍もあるが、初学者にとっては難解な内容となっている。このような状況を改善するために、ファイル処理とデータベースに関する学習支援を目的としたモデル教材の開発を行った。

このモデル教材は、大きな一つのソフトウェアとしては開発せず、複数の小規模なソフトウェアとして機能ごとに作成した。個々のプログラムが比較的単純な構造であれば、初学者にも作成が容易であり、さらに学習者が自ら応用的なプログラムに発展させることも可能と考える。

2. モデル教材の特徴

C 言語等のプログラミング言語で扱うファイルには、数種類のファイル編成法がある。ファイル編成法とは、記録媒体上にデータをどのように配置するのか、どのようにアクセスするのか等の方法を定義したものである。この中で最も単純な構造を持つファイル編成法である順編成ファイルのレコードの例を図 2.1 に示す。

フィールド 1	フィールド 2	フィールド 3	フィールド 4

図 2.1　順編成ファイルのレコードの例

ファイルを構成する最小の要素がフィールドである。例えば、氏名、住所、電話番号等の項目がフィールドに相当する。複数のフィールドを纏めて 1 件分のデータとしたものがレコードであり、コンピューターによるファイル処理は、通常レコード単位で行われる。図 2.1 の例では、1 レコードの中に 4 個のフィールドが格納されている。さらに、複数のレコードを纏めたものがファイルである。Windows 等のオペレーティングシステム（OS）では、ファイル単位で読み書き等の管理を行っている。

これに対してデータベースは、多くのデータを集積して、コンピューターを用いたデータの追加、削除、検索等の操作を行いやすい形式に分類、整理したものであり、様々な用途に利用されている。データベースの構造を大別すると「階層型」、「ネットワーク型」、「関係型」に分類される。現在、その中で一般的に利用されているのが、関係型に分類される関係データベース（リレーショナルデータベース）である。リレーショナルデータベースのテーブル（表）の例を図 2.2 に示す。

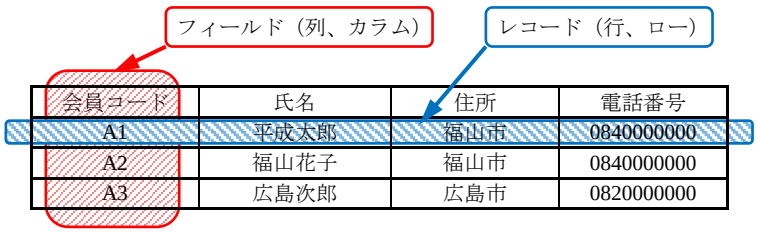

図 2.2　リレーショナルデータベースのテーブル（表）の例

リレーショナルデータベースは、関係モデル（リレーショナルデータモデル）に基づいて設計、開発されるデータベースである。データは、行と列を持ったテーブル（表）の形

式で管理される。関連する属性値を複数組み合わせて 1 件のデータに纏めたものがレコードである。レコードは、行、ローとも呼ばれる。図 2.2 の例では、会員コード、氏名、住所、電話番号が属性名、A1、平成太郎、福山市、0840000000 が属性値である。レコードを構成する個々の属性をフィールドと呼ぶ。フィールドは、列、カラムとも呼ばれる。リレーショナルデータベースでは、複数のテーブル間において、その中に含まれる同一の属性を関連付けることで、複雑かつ大規模なデータを取り扱うことを可能にしている。

本研究では、主に C 言語を用いたファイル処理における学習支援用モデル教材の開発を行うが、ファイル処理の知識、技能は、データベースと共通する箇所も少なくはなく、さらに複雑な処理を行うデータベースの理解にも役立てることが期待できる。

また、処理の動作や結果を見せるための教材としてだけでなく、プログラミングを行う際の手本として活用することを考えている。このモデル教材は、複数の小規模なソフトウェアとして機能ごとに作成しており、ツールとしての特徴を持っている。このツールとは、ある特定の機能を持った比較的小規模なソフトウェアのことであり、アプリケーションソフトウェア等の大規模なソフトウェアやシステムを作成する際に補助的に用いられたり、ユーザーが何らかの作業を行う際に支援したりする機能を持つ。ユーティリティソフトウェアと呼ばれることもある。これらの単機能で小規模なプログラムは、比較的開発しやすく、単体でのテストも容易である等の利点がある。また、ソフトウェア開発を指導する上でも、教えやすいことから有効性が期待できる。

3. 開発の環境及びプログラムの設計

モデル教材の開発に使用するコンピューター、開発言語（プログラミング言語）、実行環境、プログラムの設計等について述べる。

3.1 使用するコンピューター

モデル教材の実行は、オペレーティングシステム（OS）として Microsoft（MS）Windows シリーズが動作しているパーソナルコンピューター（パソコン）を想定している。開発環境もそれに基づいており、実際の開発には、MS Windows 7 Professional が動作し、Service Pack 1 が適用されているパソコンを使用した。また、開発後に、MS Windows 7、8.1、10 が動作するパソコンを用いてテストを行うことにより、実行環境においても正常に動作することを確認した。

3.2 開発言語（プログラミング言語）

モデル教材の開発には、授業で使用している C 言語を用いた。C 言語は、パソコンから大型のコンピューターまで幅広い分野で利用されているプログラミング言語である。多くの演算子、データ型、制御構造を持ち、構造化のサポート、移植性が高い等の特徴を持っている。コンピューターシステムの記述用に開発された経緯を持つことから、ハードウェア向けの低水準な記述も可能である。C 言語にオブジェクト指向性の拡張を施したプログラミング言語を C++言語という。C 言語と C++言語は、現在最も普及しているプログラミ

ング言語の一つである。

プログラミングに用いる C/C++ コンパイラには、Embarcadero Technologies 社が提供する Borland C++ Compiler（BCC）5.5.1 を利用する。

3.3 実行環境

開発したモデル教材を動作させるために、Windows のコマンドプロンプトを使用する。このコマンドプロンプトは、コマンドと呼ばれる命令文を用いて、プログラムの実行、Windows の操作・設定等をコマンドライン形式で行うためのツールである。cmd.exe という名前のプログラムを実行することで、コマンドラインインタプリタを起動して利用する。

モデル教材は、複数のプログラムから構成されており、コマンドプロンプトを用いて各プログラムを呼び出すことで、目的の処理を実行することができる。

3.4 プログラムの設計

ファイル処理及びプログラミングに対する学生の理解を深めるため、コマンドライン形式で実行するプログラムと、メニュー形式で実行するプログラムの 2 種類を作成した。前者は、小規模かつ単機能のプログラムを目的別に複数作成している。各プログラムには、①新規作成処理及び全削除処理、②レコードの追加処理、③レコードの挿入処理、④レコードの削除処理（1 レコードを指定）、⑤レコードの削除処理（会員番号を指定）、⑥レコードの削除処理（複数のレコード）、⑦レコードの修正・置換処理、⑧一覧表示処理、⑨一覧印刷処理と、9 種類の処理機能を持たせてあり、それぞれ単独で実行させる。後者は、メニューにより処理の目的を選択して実行するプログラムであり、処理の内訳は前者とほぼ同一である。

4. コマンドライン形式のプログラム

コマンドライン形式で実行する各プログラムの詳細を述べる。各プログラムは、「3.4 プログラムの設計」で示したように、①～⑨の 9 種類の処理機能について、それぞれ小規模なプログラムを単独で作成し実行させる。

4.1 ファイルとレコードの形式

コマンドライン形式で実行するプログラムでは、会員マスターファイルと更新ファイルの 2 種類のファイルを使用する。各ファイルの形式、役割、作成方法等について述べる。

会員マスターファイルは、プログラムの実行によって作成される順編成ファイルである。会員データとして、会員番号、氏名、電話番号が記録されており、入力された順に複数の会員データが格納されている。会員マスターファイルに記録されるレコードの形式を図 4.1 に示す。会員マスターファイルのレコードは、1 件分の会員データ（38 桁）からなり、先頭から順に会員番号（7 桁）、氏名（20 桁）、電話番号（11 桁）のフィールドを持つ。

更新ファイルの作成は、プログラムの実行では行わず、MS Windows に収録されたメモ帳等のテキストエディタを用いて行う。更新ファイルには、処理を識別するための処理コ

ード、処理対象のレコードを指定するレコード番号、各処理で必要となる会員データとして、会員番号、氏名、電話番号が記録されている。更新ファイルに記録されるレコードの形式を図 4.2 に示す。

会員番号 （7 桁）	氏名 （20 桁）	電話番号 （11 桁）

図 4.1　会員マスターファイルのレコードの形式

処理コード （2 桁）	レコード番号 （3 桁）	会員番号 （7 桁）	氏名 （20 桁）	電話番号 （11 桁）

図 4.2　更新ファイルのレコードの形式

更新ファイルのレコードは、1 件分の更新データ（43 桁）からなり、先頭から順に処理コード（2 桁）、レコード番号（3 桁）、会員番号（7 桁）、氏名（20 桁）、電話番号（11 桁）のフィールドを持つ。ただし、処理によっては不要となるフィールドがあり、その場合はダミーデータとして何も指定せずに空欄とする。各処理を単純化するため、1 つの更新ファイルには 1 件分（1 処理分）の更新データのみが記録されているものとする。

4.2　処理プログラムの詳細

各プログラムにおける処理の詳細、更新ファイルの指定方法等について述べる。プログラムは、その構造を理解しやすいように機能を単純化しており、1 つのプログラムが 1 つの処理を限定して実行する。従って、ファイル処理を実行するプログラムは、機能別に複数個を作成する。

4.2.1　新規作成処理及び全削除処理

この処理は、会員マスターファイルに対する新規作成処理及び全削除処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"N1"を指定して処理の識別を行う。残りの 41 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.3 に示す。

処理コード N1	ダミー （41 桁）

図 4.3　更新ファイルのレコード（新規作成処理及び全削除処理）

プログラムの実行においては、起動時に指定される会員マスターファイルが存在しなければ、新規作成処理として新たに会員マスターファイルを作成する。この新規作成処理後の会員マスターファイルに記録されているレコードの件数は 0 件である。一方、起動時に指定される会員マスターファイルが存在すれば、全削除処理として該当する会員マスター

ファイルのレコードをすべて削除する。この全削除後処理の会員マスターファイルに記録されたレコードの件数は 0 件となる。ただし、このプログラムでは、会員マスターファイル自体の削除は行わず、削除の必要があれば OS の機能を用いて行うものとする。

4.2.2 レコードの追加処理

この処理は、会員マスターファイルに対するレコードの追加処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"A1"を指定して処理の識別を行う。続けて、3 桁の空欄をダミーデータとして格納する。さらに、追加するための会員データを会員番号（7 桁）、氏名（20 桁）、電話番号（11 桁）の順に格納する。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.4 に示す。

処理コード A1	ダミー (3 桁)	会員番号 (7 桁)	氏名 (20 桁)	電話番号 (11 桁)

図 4.4　更新ファイルのレコード（追加処理）

プログラムの実行においては、更新ファイルの会員データを読み込み、そのレコードを既存の会員マスターファイルの最終レコードとして記録する。この追加処理後の会員マスターファイルに記録されているレコードの件数は 1 件増加する。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.3 レコードの挿入処理

この処理は、会員マスターファイルに対するレコードの挿入処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"I1"を指定して処理の識別を行う。続けて、挿入するレコードの位置を指定するためのレコード番号（3 桁）を格納する。さらに、挿入するための会員データを会員番号（7 桁）、氏名（20 桁）、電話番号（11 桁）の順に格納する。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.5 に示す。

処理コード I1	レコード番号 (3 桁)	会員番号 (7 桁)	氏名 (20 桁)	電話番号 (11 桁)

図 4.5　更新ファイルのレコード（挿入処理）

プログラムの実行においては、更新ファイルの会員データを読み込み、そのレコードを既存の会員マスターファイルの指定されたレコードの位置に記録する。その際に、元から記録されているレコードについては、ファイルの後方に順次移動させることで記録の保全を図る。この挿入処理後の会員マスターファイルに記録されているレコードの件数は 1 件増加する。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなけ

れば、エラーとして処理を中断するものとする。

4.2.4　レコードの削除処理（１レコードを指定）

この処理は、会員マスターファイルに対するレコードの削除処理を行うためのプログラムを用いて実行する。削除するレコードは 1 レコードとし、該当するレコード番号を直接指定する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"D1"を指定して処理の識別を行う。続けて、削除するレコードの位置を指定するためのレコード番号 (3 桁) を格納する。残りの 38 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.6 に示す。

処理コード D1	レコード番号 (3 桁)	ダミー (38 桁)

図 4.6　更新ファイルのレコード（削除処理、1 レコードを指定）

プログラムの実行においては、更新ファイルのレコード番号を読み込み、既存の会員マスターファイルから指定された位置のレコードを削除する。その際に、元から記録されているレコードについては、ファイルの前方に順次移動させることで、削除したレコードの空いた位置を埋める処理を行う。この削除処理（1 レコードを指定）後の会員マスターファイルに記録されているレコードの件数は 1 件減少する。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.5　レコードの削除処理（会員番号を指定）

この処理は、会員マスターファイルに対するレコードの削除処理を行うためのプログラムを用いて実行する。削除するレコードは 1 レコードとし、該当する会員番号を指定する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"D2"を指定して処理の識別を行う。続けて、3 桁の空欄をダミーデータとして格納する。さらに、削除する会員を指定するための会員番号（7 桁）を格納する。残りの 31 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.7 に示す。

処理コード D2	ダミー (3 桁)	会員番号 (7 桁)	ダミー (31 桁)

図 4.7　更新ファイルのレコード（削除処理、会員番号を指定）

プログラムの実行においては、更新ファイルの会員番号を読み込み、既存の会員マスターファイルから指定された会員のデータが記録されているレコードを削除する。その際に、元から記録されているレコードについては、ファイルの前方に順次移動させることで、削除したレコードの空いた位置を埋める処理を行う。この削除処理（会員番号を指定）後の

会員マスターファイルに記録されているレコードの件数は 1 件減少する。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.6 レコードの削除処理（複数のレコード）

この処理は、会員マスターファイルに対する複数のレコードの削除処理を行うためのプログラムを用いて実行する。削除するレコードは複数のレコードとし、削除の開始位置と終了位置を示すレコード番号を直接指定する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"D3"を指定して処理の識別を行う。続けて、削除するレコードの開始位置を指定するためのレコード番号 1（3 桁）、終了位置を指定するためのレコード番号 2（3 桁）を格納する。このレコード番号 2 のフィールドは、他の処理と桁数、役割等が異なるため、注意が必要である。残りの 35 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.8 に示す。

処理コード D3	レコード番号 1 （3 桁）	レコード番号 2 （3 桁）	ダミー （35 桁）

図 4.8　更新ファイルのレコード（削除処理、複数のレコード）

プログラムの実行においては、更新ファイルのレコード番号 1 及びレコード番号 2 を読み込み、既存の会員マスターファイルから指定された範囲にある複数のレコードを削除する。その際に、元から記録されているレコードについては、ファイルの前方に順次移動させることで、削除したレコードの空いた位置を埋める処理を行う。この削除処理（複数のレコード）後の会員マスターファイルに記録されているレコードの件数は（レコード番号 2－レコード番号 1＋1）件減少する。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.7 レコードの修正処理及び置換処理

この処理は、会員マスターファイルに対するレコードの修正処理及び置換処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"M1"を指定して処理の識別を行う。続けて、修正・置換を行うレコードの位置を指定するためのレコード番号（3 桁）を格納する。さらに、修正・置換を行うための会員データを会員番号（7 桁）、氏名（20 桁）、電話番号（11 桁）の順に格納する。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.9 に示す。

処理コード M1	レコード番号 （3 桁）	会員番号 （7 桁）	氏名 （20 桁）	電話番号 （11 桁）

図 4.9　更新ファイルのレコード（修正処理及び置換処理）

プログラムの実行においては、更新ファイルの会員データを読み込み、そのレコードを既存の会員マスターファイルの指定されたレコードの位置に上書き処理を行う。その際に、更新ファイルのいずれかのフィールドに空欄があるときは、会員マスターファイルの該当するフィールドに元から記録されているデータを利用して補い、空欄以外のフィールドのデータと合わせて上書を行う。これは、会員マスターファイルへの修正処理に相当する。一方、更新ファイルのすべてのフィールドが空欄でなければ、そのまま上書きを行う。これは、会員マスターファイルへの置換処理に相当する。これらの修正処理及び置換処理後の会員マスターファイルに記録されているレコードの件数は、いずれも処理前と同数である。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.8　一覧表示処理

この処理は、会員マスターファイルに対する一覧表示処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"L1"を指定して処理の識別を行う。残りの 41 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.10 に示す。

図 4.10　更新ファイルのレコード（一覧表示処理）

プログラムの実行においては、起動時に指定される会員マスターファイルを読み込み、リスト形式で一覧表示を行う。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

4.2.9　一覧印刷処理

この処理は、会員マスターファイルに対する一覧印刷処理を行うためのプログラムを用いて実行する。この処理に使用する更新ファイルには、処理コード（2 桁）に文字列の"P1"を指定して処理の識別を行う。残りの 41 桁はダミーデータとして、何も指定せずに空欄とする。この処理を実行するための更新ファイルに含まれるレコードの例を図 4.11 に示す。

図 4.11　更新ファイルのレコード（一覧印刷処理）

プログラムの実行においては、起動時に指定される会員マスターファイルを読み込み、リスト形式で一覧印刷を行う。なお、起動時に指定される会員マスターファイルや更新ファイルが存在しなければ、エラーとして処理を中断するものとする。

5. メニュー形式のプログラム

メニュー形式で実行するプログラムについて詳細を述べる。このプログラムは、「3.4 プログラムの設計」で示した①～⑨の 9 種類の処理機能について、統合された 1 つのメニューを作成した上で、メニューからそれぞれの処理を選択して実行する形式をとる。コマンドライン形式で用いた更新ファイルは使用せず、追加、挿入、修正、置換の各処理で必要となる会員データについては、該当する項目を実行時に入力することで対応する。各処理の基本的な構造は、コマンドライン形式のプログラムとほぼ同一であり、それらをサブプログラムとして呼び出している。このプログラムは、会員マスターファイルを保守するための簡易的な会員管理システムと言える。

5.1 ファイルとレコードの形式

メニュー形式で実行するプログラムでは、会員マスターファイルのみを使用する。このファイルは、「4.1 ファイルとレコードの形式」で示したコマンドライン形式で使用する順編成ファイルと同一の形式を持っている。

5.2 表示画面の設計

メニュー形式で実行するプログラムにおいて、実行時に表示される画面を図 5.1 に示す。1～9 の番号に 9 種類の処理を対応させて、処理メニューとして一覧表示する。そして、この番号を指定することで実行する処理の選択を行う。また、0 の番号を指定することで、このメニューを終了する。

```
処理メニュー
 1 - N1:新規作成・全削除
 2 - A1:追加
 3 - I1:挿入
 4 - D1:削除（１レコードを指定）
 5 - D2:削除（会員番号を指定）
 6 - D3:削除（複数のレコード）
 7 - M1:修正・置換
 8 - L1:一覧表示
 9 - P1:一覧印刷
 0 - E1:終了
番号を指定して処理を選択してください。
```

図 5.1　メニュー形式の表示画面

5.3 処理プログラムの詳細

メニュー形式で実行するプログラムは、メインプログラムと複数のサブプログラムから構成される。メインプログラムでは、「処理メニューの表示」、「番号の入力」、「番号に対応したサブプログラムの呼び出し」の処理を行う。サブプログラムでは、「入力画面の表示」、「レコード番号、会員データの入力」、「会員マスターファイルへの各処理」等を行う。会員マスターファイルに対するそれぞれの処理は、コマンドライン形式とほぼ同一の内容であり、それらのプログラムに対して一部変更を加えることにより、サブプログラムとして

利用している。メニュー形式で実行するプログラムにおいて、処理の概要を表した流れ図（フローチャート）を図 5.2 に示す。この流れ図では、メインプログラム及び複数のサブプログラムを代表して挿入処理を記し、他のサブプログラムについては、ほぼ同一の内容であるため省略している。

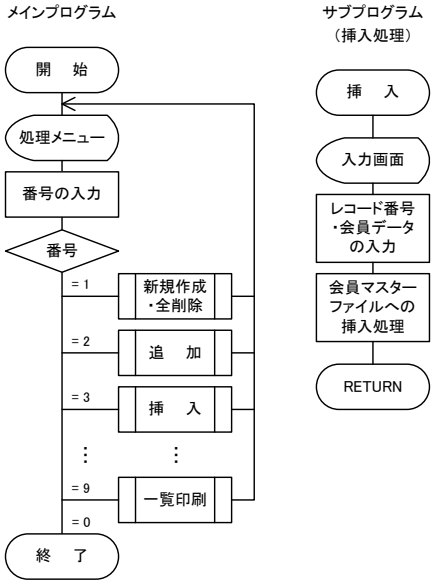

図 5.2　処理の概要を表した流れ図（挿入処理の場合）

6.　プログラムの実行

モデル教材として作成したそれぞれのプログラムについて、実際にコンピューターを用いて実行する方法を述べる。さらに、コマンドライン形式とメニュー形式のプログラムにおいて、実行時にファイルを指定する方法等について詳細を示す。

6.1　コマンドライン形式のプログラム

コマンドライン形式で実行する各プログラムは、Windows のコマンドプロンプトを使用して実行する。1 度の実行で 1 つの処理を行うことができる。起動には、処理を実行するためのプログラム、会員マスターファイル、更新ファイルの順にファイル名を指定する。コマンドライン形式で挿入処理を行う場合の指定を例として図 6.1 に示す。ここでは、処理を実行するプログラムに"fileins.exe"、会員マスターファイルに"master.dat"、更新ファイル"transact.dat"を指定している。さらに、挿入処理用の更新ファイルの内容を図 6.2 に、挿入処理を実行する前の会員マスターファイルの内容を図 6.3 に、実行した後の内容を図 6.4 に示す。挿入処理の結果、図 6.4 のレコード 3 の位置（斜線部分）に、新たなレコードが挿入されていることが分かる。

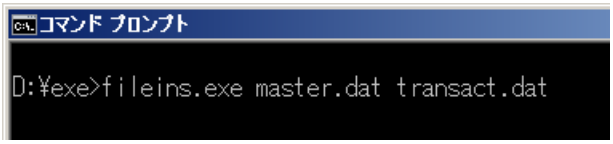

図 6.1　コマンドライン形式のプログラムの実行（挿入処理の場合）

I1	003	A000005	尾道みゆき	0848000000

図 6.2　更新ファイルの内容（挿入処理）

レコード 1	A000001	平成太郎	0840000000
レコード 2	A000002	福山花子	0840000000
レコード 3	B000011	広島次郎	0820000000

図 6.3　会員マスターファイルの内容（挿入処理の実行前）

レコード 1	A000001	平成太郎	0840000000
レコード 2	A000002	福山花子	0840000000
レコード 3	A000005	尾道みゆき	0848000000
レコード 4	B000011	広島次郎	0820000000

図 6.4　会員マスターファイルの内容（挿入処理の実行後）

6.2　メニュー形式のプログラム

メニュー形式で実行するプログラムにおいても、最初の起動には Windows のコマンドプロンプトを使用する。ただし、1 度起動した後は、メニューから番号を選択して、何度も処理を繰り返し行うことができる。起動には、処理を実行するためのプログラム、会員マスターファイルの順にファイル名を指定する。メニュー形式のプログラムを実行する例を図 6.5 に示す。ここでは、処理を実行するプログラムに"filemenu.exe"、会員マスターファイルに"master.dat"を指定している。会員マスターファイルに対する処理は、コマンドライン形式のプログラムと同様である。

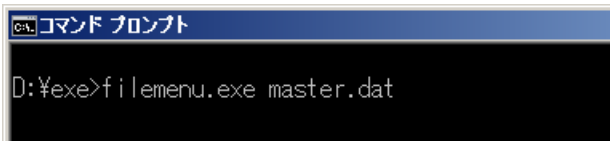

図 6.5　メニュー形式のプログラムの実行

7.　おわりに

本稿では、ファイル処理とデータベースに関する学習支援用モデル教材の開発を行った。このモデル教材に対するプログラムとしての動作、完成度等を評価するため、図 4.1 及び図 4.2 で示した形式のサンプルデータを用いて動作確認のテストを実施した。いずれのプ

ログラムにおいても、動作や実行結果に問題点等は発生せず、モデル教材のプログラムとしての完成度を確認することができた。

さらに、当初の目的であるモデル教材としての効果や有効性等を評価するため、数名の学生を対象として、本稿で開発した教材を用いたプログラミングの指導を行った。これらの学生は、情報処理推進機構（IPA）の情報処理技術者試験センターが実施する、情報処理技術者試験の基本情報技術者試験を受験するため、C 言語を用いたプログラミング技術の習得を目指している。指導では、最初にコマンドライン形式の実行プログラムを手本として示し、それぞれのプログラムの動作、ファイル等の実行結果を確認し理解させた。次に挿入処理等の一部のソースプログラムを手本として示し、処理内容を理解させた。続けて手本として示したプログラム以外のものについて作成を行わせた。いずれの学生も、事前に手本のプログラムを用いて動作や内容を確認できていたことから、比較的スムースにプログラムを完成させており、教材としての有効性がある程度確認できたように思われる。

このモデル教材では、初めに学習者に対してプログラムの動作を見せ、その後に、それをもとにプログラムを作成させることを想定している。入門書や参考書で知識を得るだけではなく、実際にパソコンでプログラムを動作させ、体感し理解することが重要であると考える。さらに、各々が工夫を加えるなどして、メニュー形式等の複雑なプログラムに発展させることができる。

このモデル教材の特徴や利点を纏めると、以下の通りである。

(1) ファイル処理の基本となるプログラムを実行して結果を確認できる。
(2) 実際にプログラムを作成する際の手本として利用できる。
(3) 小規模かつ単純なプログラムで開発することにより、プログラミングを行う学習者の理解を助けることができる。一度にすべての処理を作成しようとすると迷いやすい。
(4) 個々のプログラムを理解した後に、メニュー形式のプログラムの開発に進むと、プログラムの構造、作成の手順等を理解しやすい。また、サブプログラムの構造を把握しやすい。
(5) プログラミングの理解度が不十分な学習者においても、順を追って作成することができ、比較的容易にプログラムの作成を行うことができる。

ただし、次のような問題も見受けられた。C 言語のファイル処理を理解するためには、ポインタの理解がある程度必要である。このポインタの習得もファイル処理に劣らず難しい学習箇所であることから、ここで躓いている学生も多く、何らかの対策が必要であると考えている。

このように、これらのモデル教材を学習に用いることで、ファイル処理の理解とプログラム作成において、一定の効果があることを確認できた。このファイル処理に関する知識や技能を習得した後に、データベースに関する学習を続けて行わせることを考慮したい。リレーショナルデータベースを操作するには、一般に SQL（Structured Query Language）を用いてデータの処理を記述するため、その習得が必要となる。その学習の前段階として、表計算ソフトウェアの Excel を用いた、データベースを学習するためのモデル教材の開発

を行っている。経営学科の学生にとって、授業等で Excel を利用する機会が多いことから、データベースの理解に必要となる基本的な知識や技能について、Excel を通じて学習することにより、十分な効果が期待できると考えている。

参考文献

［1］ 佐藤真司, ツールを利用した教育支援システムの開発と評価, 福山平成大学経営学部紀要, No.10, 2014-03, pp. 33-48.
［2］ 佐藤真司, e ラーニング教材作成支援ツールの開発, 福山平成大学経営学部紀要, No.12, 2016-03, pp. 45-60.
［3］ 林春比古, 明快入門 C, ソフトバンク クリエイティブ, 2013.
［4］ 林春比古, 新訂 新 C 言語入門 ビギナー編, ソフトバンク パブリッシング, 2003.
［5］ 林春比古, 新訂 新 C 言語入門 シニア編, ソフトバンク パブリッシング, 2004.
［6］ 結城浩, 新版 C 言語プログラミングレッスン 文法編, ソフトバンク クリエイティブ, 2006.
［7］ 結城浩, 新版 C 言語プログラミングレッスン 入門編, ソフトバンク クリエイティブ, 2006.
［8］ 奥村晴彦, C 言語による最新アルゴリズム事典, 技術評論社, 1991.
［9］ 前田智美, 改訂第 3 版 Excel VBA ポケットリファレンス, 技術評論社, 2010.

Development of Model Teaching Materials to Support Learning File Processing and Database

Shinji SATO

Department of Business Administration, Faculty of Business Administration, Fukuyama Heisei University

Abstract: The purpose of this paper is to develop model teaching materials for learning file processing and database. These developments were made using widely popular C languages. Further, the author explained the outline and evaluation of these teaching materials.

Key Words: File Processing, Database, Learning Support, Model Teaching Materials, Programming, C Language