C言語プログラミング能力認定試験

1級実技試験

テーマプログラム

［第4版］

サーティファイ
情報処理能力認定委員会

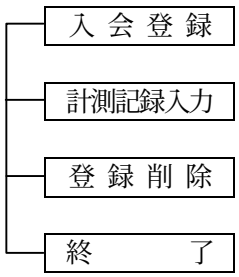
アスレチッククラブ会員管理プログラム

要 求 仕 様 書

1．目的

本システムは，アスレチッククラブにおける，メンバーの管理を目的としている。対象となるアスレチッククラブは，現状では設備の関係で，200人までの会員しか登録できないが，将来は数千人のメンバーを登録することも考えられる。

2．メニュー



3．処理説明

（1） 入会登録　　入会希望者があったとき，メンバーの空きがあれば，空いている会員コードを与えて，会員登録する。空いている番号の中では，古いものを優先する。メンバーの空きがなければ，「残念ながらただ今メンバーの空きがありません」と表示して終了する。

（2） 計測記録入力
① 毎回，運動後に計測記録を，会員コード，運動日とともに入力する。
② 入力されたデータをもとに，ある算式により，運動指数を計算する。
③ 記録は，今までの運動回数，入会時データ，最高記録データ，最新 10 回分までの運動指数である。
④ 結果の出力は，個人の履歴データと，全メンバー中の最高記録の 1 位から 10 位までの結果データを画面に表示する。

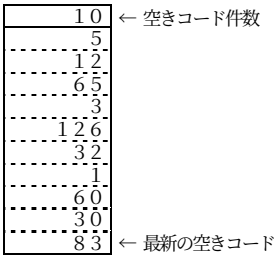
（3） 登録削除　　退会者があったとき，その計測記録を削除し，会員コードを解放する。
（退会者の履歴データは保存しない。）

4．システム詳細

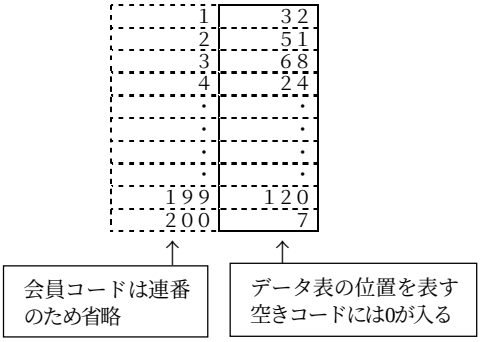
（1）ファイルイメージ

① 空きコード表：
会員コードの空きを記録した表

1レコード目は，空きコードの件数が格納され，2レコード目以降に，空きコードが古い順に格納される。



② コード・データ対照表：
会員コードから計測データ表のデータのある行を示す表



③ 計測データ表：計測記録データファイル

会員コード	A	B	C	D	E	…	N	O	P

A：計測回数　　B：初回日付　　C：初回データ
D：最高記録日付　　E：最高記録データ　　F：最新計測日付
G：最新データ　　H：1回前データ　　I：2回前データ
・・・・・　　O：8回前データ　　P：9回前データ

（2）入力データ

① 会員コード

② 運動実績

	負荷	セット	回数
運動1	W1	S1	N1
運動2	W2	S2	N2
運動3	W3	S3	N3
運動4	W4	S4	N4
運動5	W5	S5	N5

③ 負荷係数

運動1の係数 F1=0.24
運動2の係数 F2=0.36
運動3の係数 F3=0.52
運動4の係数 F4=1.05
運動5の係数 F5=2.13

④ 運動指数の計算式

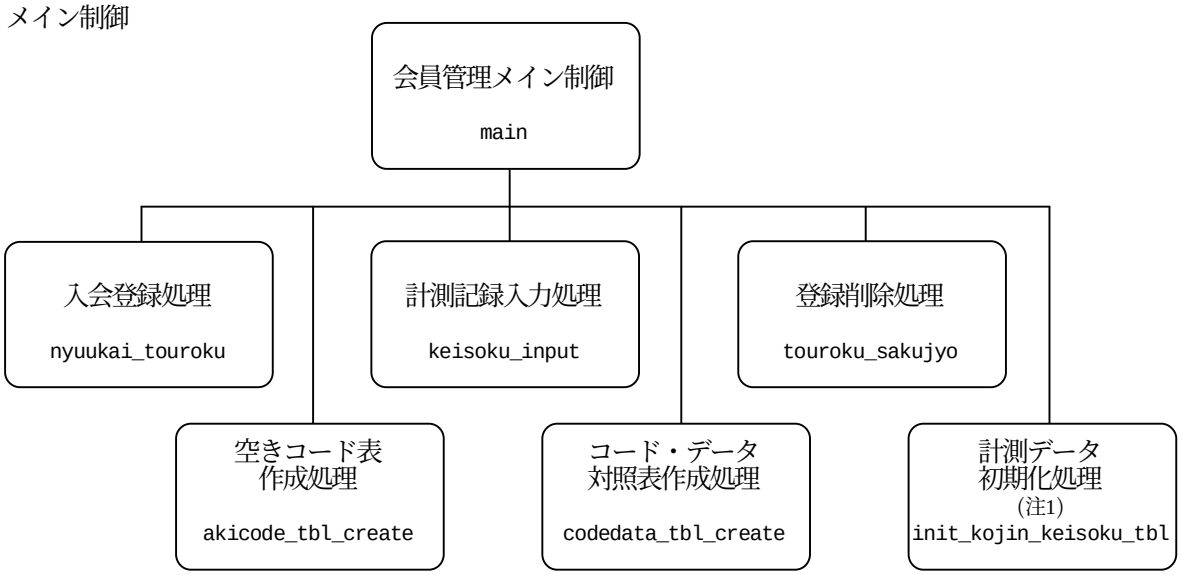
$$\sqrt{\sum_{i=1}^{5} Fi * Wi * Si * (Ni^2 / (Ni - 1))}$$

アスレチッククラブ会員管理プログラム

システム仕様書

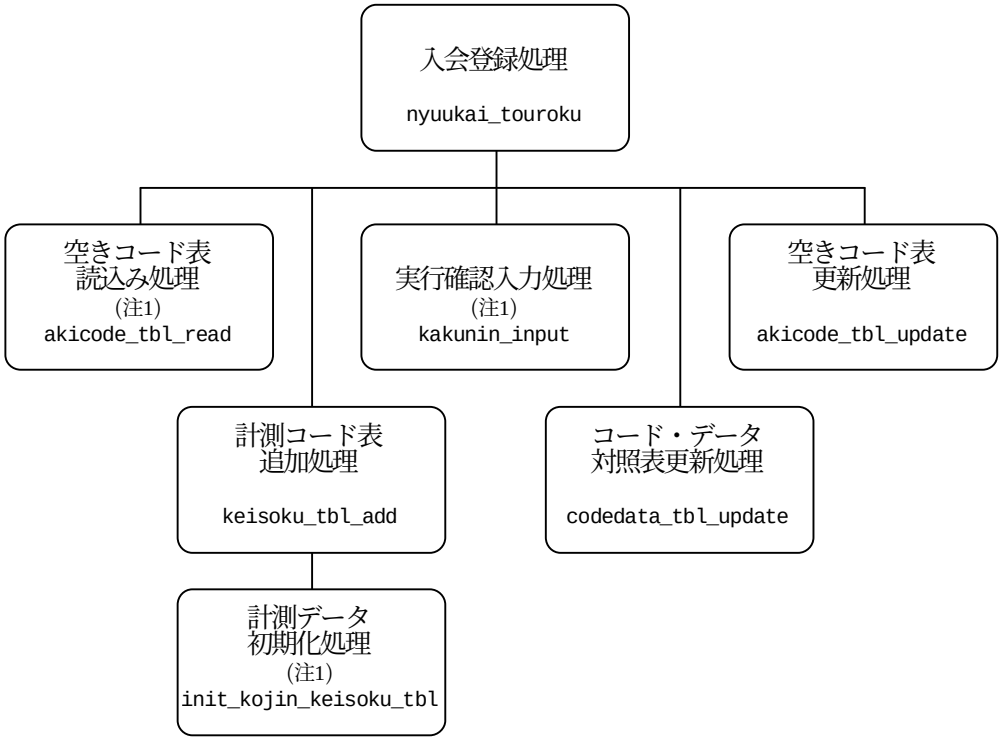
1．関数構成図

1．1．　メイン制御



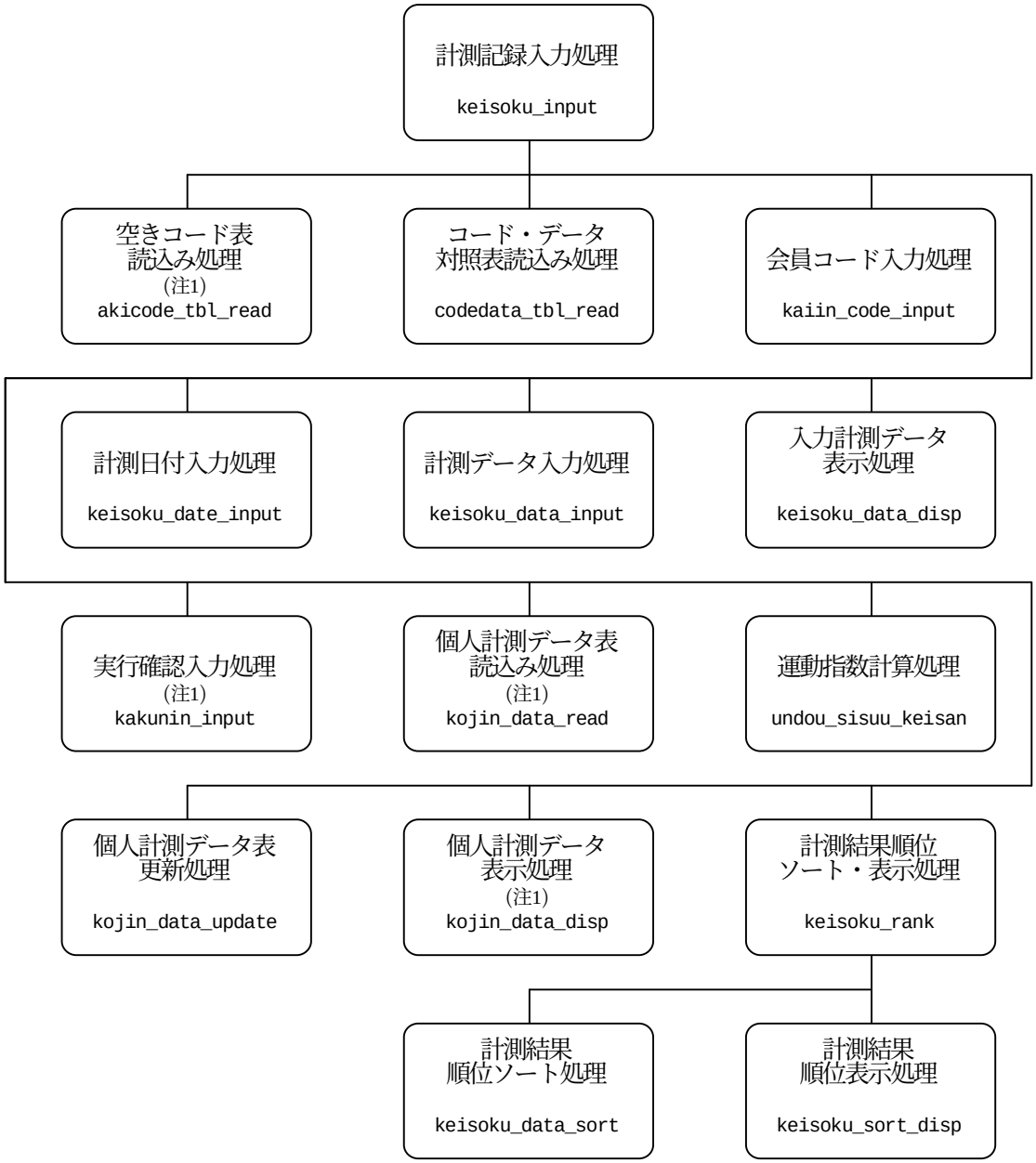
注1：共通ルーチン

1．2．　入会登録処理



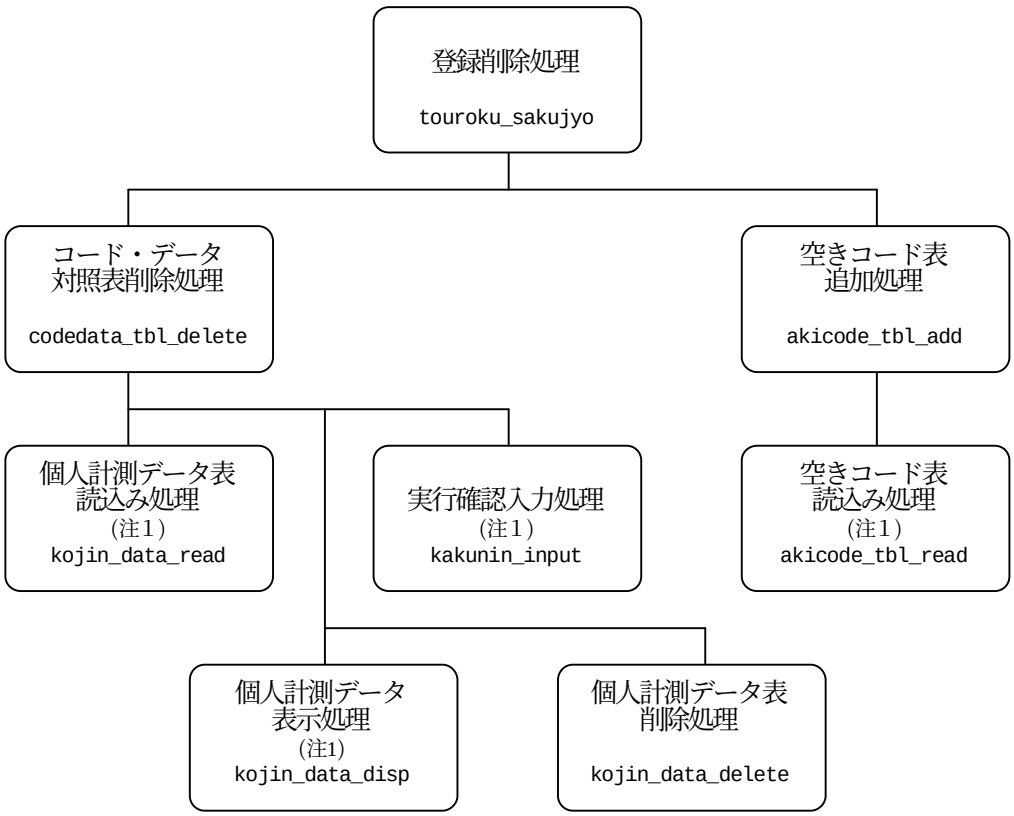
注1：共通ルーチン

1．3．　計測記録入力処理



注1：共通ルーチン

１．４．　登録削除処理



注1：共通ルーチン

2．ファイル設計

2．1．　ファイル一覧

№	ファイル名	識別子	REC数	備　考
1	空きコード表	akicode.tbl	可変	1 レコード目：件数
2	コード・データ対照表	codedata.tbl	200	
3	計測データ表	keisoku.tbl	可変	
4	計測データ表ワーク	keisoku.tmp	可変	削除処理時使用

2．1．1．空きコード表

・現時点における空き会員コードを格納する。
（1レコード目は，空きコードの件数とする。）

※ 初期状態
空きコード数　：　200
空きコード　　：　1 から 200 が順にセットされる。

№	項　目　名	フィールド名	属性	備　考
1	空きコード数	akicode_tbl	int	1 レコード目
2	空きコード	akicode_tbl	int	MAXﾚｺｰﾄﾞ数 200

2．1．2．コード・データ対照表

① 会員毎の計測データ表ファイルのレコード位置を格納する。
② 対応する会員コードは，本ファイルのレコード番号とする。
（例えば，1 レコード目には，会員コード＝1 のデータ位置がセットされる。）
③ 未登録の会員には，0 がセットされる。

※ 初期状態はすべて 0 がセットされる。

№	項　目　名	フィールド名	属性	備　考
1	計測データ表位置	codedata_tbl	int	200 レコード固定

2．1．3．計測データ表

① 会員毎の履歴データを格納する。
② 本ファイルは，会員の登録処理時に作成される。

№	項　目　名	フィールド名	属性	備　考
1	会員コード	kaiin_code	int	
2	計測回数	count	int	
3	初回日付	first_date	char[9]	YYYYMMDD
4	初回データ	first_data	int	
5	最高記録日付	max_date	char[9]	YYYYMMDD
6	最高記録データ	max_data	int	
7	最新計測日付	soku_date	char[9]	YYYYMMDD
8	計測データ	soku_data	int[10]	

※ 計測データ　　[0] ･･･ 最新データ
[1] ･･･ 1 回前データ
[2] ･･･ 2 回前データ
[3] ･･･ 3 回前データ
[4] ･･･ 4 回前データ
[5] ･･･ 5 回前データ
[6] ･･･ 6 回前データ
[7] ･･･ 7 回前データ
[8] ･･･ 8 回前データ
[9] ･･･ 9 回前データ

2．1．4．計測データ表ワーク

① 内容は，計測データ表と同様である。
② 本ファイルは，会員登録削除時にテンポラリファイルとして作成され，削除終了時に，計測データ表ファイルとして，リネームされる。

3．ファイル構成

3．1．　ヘッダファイル

№	ファイル名	概要	備　考
1	common.h	共通定義ヘッダファイル	共通定数定義，構造体定義等
2	main.h	メイン制御および共通処理ヘッダファイル	
3	nyuukai.h	入会登録処理ヘッダファイル	
4	keisoku.h	計測記録入力処理ヘッダファイル	
5	sakujyo.h	登録削除処理ヘッダファイル	

3．2．　プログラムファイル

№	ファイル名	概要	備　考
1	main.c	メイン制御関連および共通処理プログラム	共通データ領域も含む
2	nyuukai.c	入会登録処理プログラム	
3	keisoku.c	計測記録入力処理プログラム	
4	sakujyo.c	登録削除処理プログラム	

3．3．　プログラムファイルと関数の対応

3．3．1．メイン制御関連および共通処理プログラム（main.c）

- main　会員管理メイン制御
- akicode_tbl_create　空きコード表作成
- codedata_tbl_create　コード･データ対照表作成
- akicode_tbl_read　空きコード表読込み（共通プログラム）
- kakunin_input　実行確認入力（共通プログラム）
- kojin_data_read　個人計測データ表読込み（共通プログラム）
- kojin_data_disp　個人計測データ表示（共通プログラム）
- init_kojin_keisoku_tbl　計測データ初期化（共通プログラム）

3．3．2．入会登録処理プログラム（nyuukai.c）

- nyuukai_touroku　入会登録処理
- akicode_tbl_update　空きコード表更新
- keisoku_tbl_add　計測データ表追加
- codedata_tbl_update　コード･データ対照表更新

３．３．３．計測記録入力処理プログラム（keisoku.c）

・keisoku_input	計測記録入力処理
・codedata_tbl_read	コード･データ対照表読込み
・kaiin_code_input	会員コード入力
・keisoku_date_input	計測日付入力
・keisoku_data_input	計測データ入力
・keisoku_data_disp	入力計測データ表示
・undou_sisuu_keisan	運動指数計算
・kojin_data_update	個人計測データ表更新
・keisoku_rank	計測結果順位ソート･表示
・keisoku_data_sort	計測結果順位ソート
・keisoku_sort_disp	計測結果順位表示

３．３．４．登録削除処理プログラム（sakujyo.c）

・touroku_sakujyo	登録削除処理
・codedata_tbl_delete	コード･データ対照表削除
・kojin_data_delete	個人計測データ表削除
・akicode_tbl_add	空きコード表追加

4．関数定義書

4．1．　アスレチッククラブ会員管理メイン制御

4．1．1．会員管理メイン制御

書式	int　main(void)
パラメータ	なし
戻り値	なし
処理概要	・アスレチッククラブの会員管理処理のメイン制御を行う。 ・初期処理として空きコード表，コード・データ対照表を読込みモードでOPENした後に，エラーが発生した場合，各ファイルが存在しないとみなして新規に作成する。 また，このとき，作成中メッセージを表示する。

4．1．2．空きコード表作成

書式	int　akicode_tbl_create(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：空きコード表ファイルI/Oエラー
処理概要	・空きコード表ファイルを書込みモードでOPENする。 ・空きコードテーブルに以下の値をセットする。 空きコードテーブル[0] ＝ 200(空きコード数) 空きコードテーブル[1] ＝ 1 空きコードテーブル[2] ＝ 2 … 空きコードテーブル[200] ＝ 200 ・空きコード表ファイルに空きコードテーブルを書き込む。

4．1．3．コード･データ対照表作成

書式	int　codedata_tbl_create(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：コード・コード対照表ファイルI/Oエラー
処理概要	・コード・データ対照表ファイルを書込みモードでOPENする。 ・コード・データ対照表テーブルをNULLクリアする。 ・コード・データ対照表ファイルに書き込む。

4．1．4．計測データ初期化 (共通)

書式	struct KEISOKU_TBL init_kojin_keisoku_tbl(void)
パラメータ	なし
戻り値	初期化された状態の計測データ
処理概要	初期化された状態の計測データを返す。

4．2．　入会登録処理

4．2．1．入会登録

書式	int　　nyuukai_touroku(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：入会登録処理異常終了
処理概要	・空きコード表ファイルを参照し，会員の空きコードが存在する場合，入会登録処理を行う。空きコードがない場合，残念メッセージを表示してリターンする。 ・入会登録処理 1．空きコード表から登録した会員コードを削除する。 2．計測データ表にレコードを追加する。 3．コード・データ対照表に，追加した計測データ表のレコード位置をセットし，更新する。

4．2．2．空きコード表読込み（共通）

書式	int　　akicode_tbl_read(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：空きコード表ファイルI/Oエラー
処理概要	・空きコード表ファイルを読込みモードでOPENする。 ・空きコード表ファイルをEOFになるまで空きコード表に読み込む。

4．2．3．実行確認入力（共通）

書式	int　　kakunin_input(char *msg)
パラメータ	char　　*msg　　：確認メッセージ内容
戻り値	OK(0)　：Yes 入力　･･･ 処理実行 NG(-1)　：No　入力　･･･ 処理中止
処理概要	・渡された確認メッセージを表示し，Y/Nの入力を行う。

4．2．4．空きコード表更新

書式	int　　akicode_tbl_update(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：空きコード表ファイルI/Oエラー
処理概要	・空きコード表ファイルを書込みモードでOPENする。 （ファイルが存在する場合，内容を破棄する。） ・空きコード表から登録会員コードを削除(編集)し，空きコード表ファイルに書き込む。 ・空きコード表編集 1．2 レコード目のデータを削除し，データを詰める。 2．1 レコード目の空きコード数をデクリメントする。

4．2．5．計測データ表追加

書式	int　keisoku_tbl_add(long *fptr, int kaiin_code)
パラメータ	long　*fptr　　：計測データ表の追加書込みを行ったファイル位置(出力) int　kaiin_code　：登録会員コード
戻り値	OK(0)　：正常終了 NG(-1)　：計測データ表ファイルI/Oエラー
処理概要	・計測データ表ファイルを追加モードでOPENする。 ・計測データ表ファイルの最後に 1 レコードを追加する。 ・追加したファイルポインタを求め，fptrにセットする。

4．2．6．コード・データ対照表更新

書式	int　codedata_tbl_update(int kaiin_code, long fptr)
パラメータ	int　kaiin_code　：登録会員コード long　fptr　　：計測データ表の追加書込みを行ったファイル位置
戻り値	OK(0)　：正常終了 NG(-1)　：コード・データ対照表ファイルI/Oエラー
処理概要	・コード・データ対照表ファイルを読書きモードでOPENする。 ・コード・データ対照表テーブルに読み込む。 ・登録する会員コードをインデックスとして，fptrより計測データのレコード番号を求め，対応するコード・データ対照表にセットする。 レコード番号 ＝（fptr ／ 計測データ表レコード長）＋ 1 fptrが 0 から始まるため 1 を加算する。 ・セット終了後，ファイルポインタを先頭に戻し，ファイルに書き込む。

4．3．　計測記録入力処理

4．3．1．計測記録入力

書式	int　　keisoku_input(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：計測記録入力処理異常終了
処理概要	・計測記録入力処理のメイン制御を行う。 ・空きコード表の読込みを行い，登録会員の存在チェックを行う。 ・コード・データ対照表を読み込む。 ・計測情報データを入力し運動指数を求め，個人情報及び計測結果の上位10人までのデータを表示する。

4．3．2．コード・データ対照表読込み

書式	int　　codedata_tbl_read(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：コード・データ対照表ファイルI/Oエラー
処理概要	・コード・データ対照表ファイルを読込みモードでOPENする。 ・コード・データ対照表の読込みを行う。

4．3．3．会員コード入力

書式	void　　kaiin_code_input(int *kaiin_code)
パラメータ	int　　*kaiin_code　　：入力会員コード(出力)
戻り値	なし
処理概要	・会員コードの入力を行い，以下のチェックを行う。 1．ニューメリック・チェック 2．範囲チェック(0＜会員コード≦200) 3．会員コード登録チェック

4．3．4．計測日付入力

書式	void　　keisoku_date_input(char *keisoku_date)
パラメータ	char　　*keisoku_date　：入力計測日付(出力)
戻り値	なし
処理概要	・計測日付の入力を行い，以下のチェックを行う。 1．入力桁数チェック(YYYYMMDD：8 桁) 2．ニューメリック・チェック 3．月範囲チェック(1≦月≦12) 4．日範囲チェック(1≦日≦31)

4．3．5．計測データ入力

書式	void　keisoku_data_input(int idx)
パラメータ	int　idx　：運動種別インデックス
戻り値	なし
処理概要	・負荷，セット数，回数の入力を行い，以下のチェックを行う。 1．入力桁数チェック(3 桁以上の場合エラー) 2．ニューメリック･チェック 3．上限値範囲チェック(1≦入力値≦100) (ただし，セット数・回数のときのみ) ・入力データをテーブルにセットする。

4．3．6．入力計測データ表示

書式	void　keisoku_data_disp(int kaiin_code, char *keisoku_date)
パラメータ	int　kaiin_code　：会員コード char　*keisoku_date　：計測日付
戻り値	なし
処理概要	・入力された計測データの表示を行う。

4．3．7．運動指数計算

書式	void　undou_sisuu_keisan(int *undou_sisuu)
パラメータ	int　*undou_sisuu　：運動指数算出結果
戻り値	なし
処理概要	・入力された計測データをもとに運動指数の計算を行う。 ・計算式を以下に示す。(Ni≦1の場合，計算対象外とする。) $\sqrt{\sum_{i=1}^{5} Fi * Wi * Si * (Ni^2 / (Ni - 1))}$ Fi：負荷係数（ 0.24，0.36，0.52，1.05，2.13 ） Wi：入力負荷（MAX 999） Si：セット数（MAX 100） Ni：運動回数（MAX 100）

4．3．8．個人計測データ表更新

書式	int　kojin_data_update(int kaiin_code, char *keisoku_date, int undou_sisuu)
パラメータ	int　kaiin_code　：会員コード char　*keisoku_date　：計測日付 int　undou_sisuu　：運動指数
戻り値	OK(0)　：正常終了 NG(-1)　：計測データ表ファイルI/Oエラー
処理概要	・計測データ表に該当データをセットする。 ・1 回目　：初回データ，最高記録データ，最新データにセットする。 ・2 回目以降　：最高記録の場合，最高記録データにセットする。 履歴データを移動し，最新データをセットする。 ・計測データ表ファイルを読書きモードでOPENする。 ・ファイルポインタを該当レコード位置にシークする。 ・計測データ表ファイルの書込みを行う。

4．3．9．計測結果順位ソート・表示

書式	int　keisoku_rank(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：計測データ表ファイルI/Oエラー
処理概要	・計測データ表を読込みモードでOPENする。 ・計測データ表のすべてのレコードを読み込み，ソート用テーブルにセットする。 ・計測データが存在する会員の件数を求める。(計測回数が 0 以上) ・セットしたテーブルを最高記録データでソートし，上位 10 人までの結果を表示する。

4．3．１０．計測結果順位ソート

書式	void　keisoku_data_sort(int cnt)
パラメータ	int　cnt　：計測データ件数
戻り値	なし
処理概要	・計測データのソート用テーブルを計測データ件数分，最高記録データでソートする。

4．3．１１．計測結果順位表示

書式	void　keisoku_sort_disp(int cnt)
パラメータ	int　cnt　：計測データ件数
戻り値	なし
処理概要	・ソートされた計測データのソート用テーブルの上位 10 人までの会員コード，最高記録データを表示する。 ・10 人に満たない場合は，計測データ件数分表示する。

4．3．１2．個人計測データ表読込み (共通)

書式	int　kojin_data_read(int kaiin_code)
パラメータ	int　kaiin_code ：会員コード
戻り値	OK(0)　：正常終了 NG(-1)　：計測データ表ファイルI/Oエラー
処理概要	・計測データ表ファイルを読込みモードでOPENする。 ・会員のデータ位置までシークする。 　データ位置　＝ 　（コード・データ対照表[会員コード　－　1]　－　1） 　× 計測データ表レコード長 ・計測データ表の読込みを行う。

4．3．１3．個人計測データ表示 (共通)

書式	void　kojin_data_disp(int kaiin_code, char *msg)
パラメータ	int　kaiin_code ：表示会員コード char　*msg　：表示メッセージ
戻り値	なし
処理概要	・計測データ表の個人データを表示する。 ・表示項目 　1．会員コード 　2．計測回数 　3．初回日付 　4．初回データ 　5．最高記録日付 　6．最高記録データ 　7．最新計測日付 　8．最新計測データ 　9．計測データ(1 回前～9 回前)

４．４．　登録削除処理

４．４．１．登録削除

書式	int　　touroku_sakujyo(void)
パラメータ	なし
戻り値	OK(0)　：正常終了 NG(-1)　：登録削除処理 異常終了
処理概要	・退会する会員コードを入力し，該当する会員コードのデータを更新する。 ・登録削除処理 １．コード・データ対照表に，削除する計測データ表のレコード位置を 0 にクリアする。 ２．計測データ表から該当レコードを削除する。 ３．空きコード表に退会する会員コードを追加する。

４．４．２．コード・データ対照表削除

書式	int　　codedata_tbl_delete(int kaiin_code)
パラメータ	int　　kaiin_code　：退会会員コード
戻り値	OK(0)　　：正常終了 CANCEL(1)　：処理中止 NG(-1)　　：コード・データ対照表ファイルI/Oエラー 計測データ表ファイルI/Oエラー
処理概要	・コード・データ対照表ファイルを読書きモードでOPENする。 ・コード・データ対照表ファイルを読み込み，退会する会員の計測データ表レコード位置を取得する。 ・取得したレコード位置で計測データ表を読み込み，退会者データを表示する。（kojin_data_read, kojin_data_disp を呼び出す。） ・表示したデータを削除してよいかの確認後，計測データ表から該当レコードを削除する。（kakunin_input, kojin_data_delete を呼び出す。） ・削除が正常に終了したら，コード・データ対照表の該当位置に 0 をセットして，更新する。

４．４．３．空きコード表追加

書式	int　　akicode_tbl_add(int kaiin_code)
パラメータ	int　　kaiin_code　：退会会員コード
戻り値	OK(0)　：正常終了 NG(-1)　：空きコード表ファイルI/Oエラー
処理概要	・空きコード表の読込みを行う。 （akicode_tbl_read　を呼び出す。） ・空きコード表ファイルを書込みモードでOPENする。 （ファイルが存在する場合，内容を破棄する。） ・空きコード表に退会会員コードを追加(編集)し，空きコード表ファイルに書き込む。 ・空きコード表編集 １．ファイルの最後に退会会員コードを追加する。 ２．1 レコード目の空きコード数をインクリメントする。

４．４．４．個人計測データ表削除

書式	int　　kojin_data_delete（ int kaiin_code ）
パラメータ	int　　kaiin_code ：退会会員コード
戻り値	OK(0)　：正常終了 NG(-1)　：計測データ表ファイルI/Oエラー
処理概要	・計測データ表ファイルを読込みモードでOPENする。 ・ワークファイルを書込みモードでOPENする。 ・計測データ表をEOFになるまで読み込み，退会会員のデータ以外をワークファイルに書き込む。 ・各ファイルをCLOSEし，計測データ表ファイルを削除する。 ・削除後に，ワークファイルを計測データ表ファイルにリネームする。

アスレチッククラブ会員管理プログラム

ソースプログラムリスト

```
/************************************/
/*  common.h                        */
/*    ｷｮｳﾂｳ ﾍｯﾀﾞ ﾌｧｲﾙ                */
/************************************/

#define MEMBER_MAX          200              /* ﾒﾝﾊﾞｰｽｳ MAX   */
#define AKICODE_TBL_NAME    "akicode.tbl"    /* ｱｷｺｰﾄﾞﾋｮｳ ﾌｧｲﾙ ﾒｲ*/
#define CODEDATA_TBL_NAME   "codedata.tbl"   /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ﾒｲ*/
#define KEISOKU_TBL_NAME    "keisoku.tbl"    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾒｲ */

#define  TRUE        1      /* ｼﾝ            */
#define  FALSE       0      /* ｷﾞ            */
#define  OK          0      /* ｾｲｼﾞｮｳ        */
#define  CANCEL      1      /* ｼｮﾘ ﾁｭｳｼ      */
#define  NG          -1     /* ｲｼﾞｮｳ         */

/* ｹｲｿｸ ﾃﾞｰﾀ ﾃｰﾌﾞﾙ */
struct KEISOKU_TBL {
    int    kaiin_code;          /* ｶｲｲﾝ ｺｰﾄﾞ        */
    int    count;               /* ｹｲｿｸ ｶｲｽｳ        */
    char   first_date[ 9 ];     /* ｼｮｶｲ ﾋﾂﾞｹ        */
    int    first_data;          /* ｼｮｶｲ ﾃﾞｰﾀ        */
    char   max_date[ 9 ];       /* ｻｲｺｳ ｷﾛｸ ﾋﾂﾞｹ    */
    int    max_data;            /* ｻｲｺｳ ｷﾛｸ ﾃﾞｰﾀ    */
    char   soku_date[ 9 ];      /* ｻｲｼﾝ ｹｲｿｸ ﾋﾂﾞｹ   */
    int    soku_data[ 10 ];     /* ｹｲｿｸ ﾃﾞｰﾀ        */
};

/* ﾆｭｳﾘｮｸ ｹｲｿｸ ﾃﾞｰﾀ */
struct KEISOKU_INPUT {
    int    huka;                /* ﾌｶ   */
    int    set;                 /* ｾｯﾄ  */
    int    kaisuu;              /* ｶｲｽｳ */
};
```

```
/************************************/
/*  main.h                          */
/*    ﾒｲﾝ ｼｮﾘ ﾍｯﾀﾞ ﾌｧｲﾙ             */
/************************************/

static int  codedata_tbl_create( void );
static int  akicode_tbl_create( void );

int  akicode_tbl_read( void );
int  kakunin_input( char *msg );
int  kojin_data_read( int kaiin_code );
void kojin_data_disp( int kaiin_code, char *msg );
struct KEISOKU_TBL init_kojin_keisoku_tbl( void );
```

```
/************************************/
/*  nyuukai.h                       */
/*    ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ ﾍｯﾀﾞ ﾌｧｲﾙ          */
/************************************/

int  nyuukai_touroku( void );

static int  akicode_tbl_update( void );
static int  keisoku_tbl_add( long *fptr, int kaiin_code );
static int  codedata_tbl_update( int kaiin_code, long fptr );
```

```
/************************************/
/*  keisoku.h                       */
/*    ｹｲｿｸｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ ﾍｯﾀﾞ ﾌｧｲﾙ      */
/************************************/

int  keisoku_input( void );

static int  codedata_tbl_read( void );
static void kaiin_code_input( int *kaiin_code );
static void keisoku_date_input( char *keisoku_date );
static void keisoku_data_input( int idx );
static void keisoku_data_disp( int kaiin_code, char *keisoku_date );
static void undou_sisuu_keisan( int *undou_sisuu );
static int  kojin_data_update( int kaiin_code, char *keisoku_date, int undou_sisu
u );
static int  keisoku_rank( void );
static void keisoku_data_sort( int cnt );
static void keisoku_sort_disp( int cnt );
```

```
/************************************/
/*  sakujyo.h                       */
/*    ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ ﾍｯﾀﾞ ﾌｧｲﾙ          */
/************************************/

int  touroku_sakujyo( void );

static int  codedata_tbl_delete( int kaiin_code );
static int  kojin_data_delete( int kaiin_code );
static int  akicode_tbl_add( int kaiin_code );
```

```
/************************************/
/*  main.c                          */
/*       ｱｽﾚﾁｯｸ ｸﾗﾌﾞ ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ     */
/************************************/
#include <stdio.h>
#include <string.h>

#include "common.h"
#include "main.h"
#include "nyuukai.h"
#include "keisoku.h"
#include "sakujyo.h"


/************************************/
/* ｷｮｳﾂｳ ﾃﾞｰﾀ                       */
/************************************/

/* ｱｷｺｰﾄﾞ ﾋｮｳ */
int akicode_tbl[ MEMBER_MAX + 1 ];

/* ｺｰﾄﾞﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ */
int codedata_tbl[ MEMBER_MAX ];

/* ｺｼﾞﾝﾍﾞﾂ ｹｲｿｸ ﾃﾞｰﾀﾋｮｳ */
struct KEISOKU_TBL kojin_keisoku_tbl;

/* ｿｰﾄ ﾖｳ ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ */
struct KEISOKU_TBL sort_keisoku_tbl[ MEMBER_MAX ];


/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ               */
/*  ﾒｲﾝ                             */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                    */
/*   ﾘﾀｰﾝ   : ﾅｼ                    */
/************************************/
int main( void )
{
    int    loop = TRUE;                    /* ﾙｰﾌﾟ ﾌﾗｸﾞ                */
    char   work[ 128 ];                    /* ﾆｭｳﾘｮｸ ﾜｰｸ               */
    FILE   *fp;                            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ                */
    char   *fname1 = AKICODE_TBL_NAME;     /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ          */
    char   *fname2 = CODEDATA_TBL_NAME;    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ */
    int    i;                              /* ｲﾝﾃﾞｯｸｽ                   */

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname1, "rb" )) == NULL ) {
        printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ ﾇ ｻｸｾｲ ｼﾃｲﾏｽ｡" );

        /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ ｻｸｾｲ */
        akicode_tbl_create( );
    }
    else {
        /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname2, "rb" )) == NULL ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ﾇ ｻｸｾｲ ｼﾃｲﾏｽ｡" );

        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ｻｸｾｲ */
        codedata_tbl_create( );
    }
    else {
        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
    }

    while( loop ) {
```

```
        /* ﾃｰﾌﾞﾙ ｼｮｷ ｸﾘｱ */
        akicode_tbl[ 0 ] = 0;
        for( i = 0; i < MEMBER_MAX; i++) {
            akicode_tbl[ i + 1 ] = 0;
            codedata_tbl[ i ] = 0;
            sort_keisoku_tbl[ i ] = init_kojin_keisoku_tbl();
        }

        kojin_keisoku_tbl = init_kojin_keisoku_tbl();
        printf( "¥n" );
        printf( "¥n ******************************" );
        printf( "¥n  ｱｽﾚﾁｯｸ ｸﾗﾌﾞ ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ " );
        printf( "¥n ******************************" );
        printf( "¥n ｼｮﾘ ヲ ｾﾝﾀｸ ｼﾃｸﾀﾞｻｲ" );
        printf( "¥n 1:ﾆｭｳｶｲ ﾄｳﾛｸ" );
        printf( "¥n 2:ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ" );
        printf( "¥n 3:ﾄｳﾛｸ ｻｸｼﾞｮ" );
        printf( "¥n E:ｼｭｳﾘｮｳ" );
        printf( "¥n ? " );

        /* ｼｮﾘｸﾌﾞﾝ ﾆｭｳﾘｮｸ */
        work[ 0 ] = '¥0';
        scanf( "%s", work );

        /* ﾆｭｳﾘｮｸ ｹﾀｽｳ ﾁｪｯｸ -> 1 ｲｶﾞｲ ? */
        if( strlen( work ) != 1 ) {
            printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
            continue;
        }

        switch( work[ 0 ] ) {
            case '1':       /* ﾆｭｳｶｲ ﾄｳﾛｸ     */
                nyuukai_touroku( );
                break;

            case '2':       /* ｹｲｿｸ ｷﾛｸ ｼｭｳｹｲ */
                keisoku_input( );
                break;

            case '3':       /* ﾄｳﾛｸ ｻｸｼﾞｮ     */
                touroku_sakujyo( );
                break;

            case 'e':       /* ｼｭｳﾘｮｳ         */
            case 'E':
                loop = FALSE;
                break;

            default:
                printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
                break;
        }
    }
    return OK;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                   */
/*   ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ ｻｸｾｲ           */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                     */
/*   ﾘﾀｰﾝ   : 0:OK                   */
/*            -1:NG                  */
/************************************/
static int akicode_tbl_create( void )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ        */
    int     i;                          /* ｲﾝﾃﾞｯｸｽ          */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ       */
```

```
    char    *fname = AKICODE_TBL_NAME;    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ */

    /* ｱｷ ｺｰﾄﾞ ﾍﾝｼｭｳ */
    akicode_tbl[ 0 ] = MEMBER_MAX;
    for( i = 1; i < MEMBER_MAX + 1; i++ ) {
        akicode_tbl[ i ] = i;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "w+b" )) == NULL ) {
        printf( "¥n ｱｷ ｺｰﾄﾞﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)akicode_tbl, sizeof( akicode_tbl ), 1, fp ) )
            != 1 ) {
        printf( "¥n ｱｷ ｺｰﾄﾞﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                  */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                    */
/*   ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ｻｸｾｲ    */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                      */
/*   ﾘﾀｰﾝ   : 0:OK                    */
/*            -1:NG                   */
/************************************/
static int codedata_tbl_create( void )
{
    int     ret;                            /* ﾘﾀｰﾝ ｺｰﾄﾞ                */
    FILE    *fp;                            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ               */
    char    *fname = CODEDATA_TBL_NAME;     /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ */
    int     i;                              /* ｲﾝﾃﾞｯｸｽ                   */

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "w+b" )) == NULL ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    for( i = 0; i < MEMBER_MAX; i++ )
        codedata_tbl[ i ] = 0;

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)codedata_tbl, sizeof( codedata_tbl ), 1, fp ) )
            != 1 ) {
        /* WRITE ｴﾗｰ */
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
```

```
}


/************************************/
/* ｷｮｳﾂｳ ﾙｰﾁﾝ                        */
/************************************/


/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                  */
/*   ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ READ          */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                     */
/*   ﾘﾀｰﾝ   : 0:OK                   */
/*           -1:NG                  */
/************************************/
int akicode_tbl_read( void )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ         */
    int     i;                          /* ｲﾝﾃﾞｯｸｽ           */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ        */
    char    *fname = AKICODE_TBL_NAME;/* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ */

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "rb" )) == NULL ) {
        printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    for( i = 0; i < MEMBER_MAX + 1; i++ ) {
        /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ READ -> 1 ｲｶﾞｲ ? */
        if( (ret = fread( (char *)&akicode_tbl[ i ], sizeof( int ), 1, fp ) )
                != 1 ) {
            /* READ ｴﾗｰ ｱﾘ ? */
            if( ferror( fp ) != 0 ) {
                printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                ret = NG;
            }
            else {
                /* ﾌｧｲﾙ EOF ﾃﾞ ﾅｲ ? */
                if( feof( fp ) == 0 ) {
                    printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                    ret = NG;
                }
                else {
                    ret = OK;
                }
            }
            break;
        }
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}


/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                */
/*  ﾆｭｳﾘｮｸ ｶｸﾆﾝ                     */
/*   ﾊﾟﾗﾒｰﾀ : ｶｸﾆﾝ ﾒｯｾｰｼﾞ             */
/*   ﾘﾀｰﾝ   : 0:OK                   */
/*           -1:NG                  */
/************************************/
int kakunin_input( char *msg )
{
    int     ret;                /* ﾘﾀｰﾝ ｺｰﾄﾞ    */
```

```
    int     loop = TRUE;        /* ﾙｰﾌﾟ ﾌﾗｸﾞ     */
    char    work[ 128 ];        /* ﾆｭｳﾘｮｸ ﾜｰｸ    */

    while( loop ) {
        /* ｶｸﾆﾝ ﾋｮｳｼﾞ */
        printf( msg );
        printf( "¥n ? " );

        /* Y/N ﾆｭｳﾘｮｸ */
        work[ 0 ] = '¥0';
        scanf( "%s", work );

        /* ﾆｭｳﾘｮｸ ｹﾀｽｳ ﾁｪｯｸ -> 1 ｲｶﾞｲ ? */
        if( strlen( work ) != 1 ) {
            printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
            continue;
        }

        switch( work[ 0 ] ) {
            case 'Y':   /* Yes */
            case 'y':
                ret  = OK;
                loop = FALSE;
                break;

            case 'N':   /* No */
            case 'n':
                ret  = NG;
                loop = FALSE;
                break;

            default:
                printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
                break;
        }
    }

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                  */
/*   ｺｼﾞﾝ ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ     */
/*   ﾊﾟﾗﾒｰﾀ : ｶｲｲﾝｺｰﾄﾞ                  */
/*   ﾘﾀｰﾝ   : 0:OK                    */
/*            -1:NG                   */
/************************************/
int kojin_data_read( int kaiin_code )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ                */
    FILE    *fp;                        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾎﾟｲﾝﾀ */
    long    fptr;                       /* ｹｲｿｸ ﾃﾞｰﾀ ﾌｧｲﾙ ﾎﾟｲﾝﾀ     */
    char    *fname = KEISOKU_TBL_NAME;/* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ       */

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "rb" )) == NULL ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｶﾞｲﾄｳ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ ｾｯﾄ */
    fptr = ( codedata_tbl[ kaiin_code - 1 ] - 1 ) *
           sizeof( struct KEISOKU_TBL );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾖ ﾀｲｼｮｳ ﾉ ｲﾁ ﾏﾃﾞ SEEK -> OK ? */
    if( (ret = fseek( fp, fptr, SEEK_SET )) != OK ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ SEEK ｴﾗｰ" );

        /* ｹｲｿｸ ﾃﾞｰﾀ  ﾋｮｳ ﾌｧｲﾙ CLOSE */
```

```
        fclose( fp );
        return NG;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ -> 1ｲｶﾞｲ ? */
    if( (ret = fread( (char *)&kojin_keisoku_tbl, sizeof( kojin_keisoku_tbl ),
              1, fp )) != 1 ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ READ ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ  ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*   ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳｼﾞ                 */
/*   ﾊﾟﾗﾒｰﾀ : ｶｲｲﾝｺｰﾄﾞ                 */
/*            ﾋｮｳｼﾞ ﾒｯｾｰｼﾞ            */
/*   ﾘﾀｰﾝ   : ﾅｼ                       */
/************************************/
void kojin_data_disp( int kaiin_code, char *msg )
{
    printf( msg );
    printf( "¥n ｶｲｲﾝｺｰﾄﾞ %3d", kaiin_code );

    if( kojin_keisoku_tbl.count != 0 ) {
        printf( "¥n¥n ｹｲｿｸｶｲｽｳ   ｼｮｶｲ ﾋﾂﾞｹ   ﾃﾞｰﾀ     " );
        printf( "ｻｲｺｳ ﾋﾂﾞｹ   ﾃﾞｰﾀ     ｻｲｼﾝ ﾋﾂﾞｹ   ﾃﾞｰﾀ" );

        printf( "¥n   %3d", kojin_keisoku_tbl.count );

        printf( "   %4.4s-%2.2s-%2.2s",
           &kojin_keisoku_tbl.first_date[ 0 ],
           &kojin_keisoku_tbl.first_date[ 4 ],
           &kojin_keisoku_tbl.first_date[ 6 ] );

        printf( "  %4d", kojin_keisoku_tbl.first_data );

        printf( "   %4.4s-%2.2s-%2.2s",
           &kojin_keisoku_tbl.max_date[ 0 ],
           &kojin_keisoku_tbl.max_date[ 4 ],
           &kojin_keisoku_tbl.max_date[ 6 ] );

        printf( "  %4d", kojin_keisoku_tbl.max_data );

        printf( "   %4.4s-%2.2s-%2.2s",
           &kojin_keisoku_tbl.soku_date[ 0 ],
           &kojin_keisoku_tbl.soku_date[ 4 ],
           &kojin_keisoku_tbl.soku_date[ 6 ] );

        printf( "  %4d", kojin_keisoku_tbl.soku_data[ 0 ] );

        printf( "¥n¥n   1ｶｲﾏｴ  2ｶｲﾏｴ  3ｶｲﾏｴ  4ｶｲﾏｴ  ");
        printf( "5ｶｲﾏｴ  6ｶｲﾏｴ  7ｶｲﾏｴ  8ｶｲﾏｴ  9ｶｲﾏｴ" );

        printf( "¥n " );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 1 ] );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 2 ] );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 3 ] );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 4 ] );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 5 ] );
        printf( "   %4d", kojin_keisoku_tbl.soku_data[ 6 ] );
```

```
        printf( "    %4d", kojin_keisoku_tbl.soku_data[ 7 ] );
        printf( "    %4d", kojin_keisoku_tbl.soku_data[ 8 ] );
        printf( "    %4d", kojin_keisoku_tbl.soku_data[ 9 ] );
    }
    else {
        printf( "    ｹｲｿｸ ﾃﾞｰﾀ ｶﾞ ｱﾘﾏｾﾝ" );
        return;
    }
    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*   ｹｲｿｸ ﾃﾞｰﾀ ｼｮｷｶ                    */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                        */
/*   ﾘﾀｰﾝ   : ｹｲｿｸ ﾃﾞｰﾀ                 */
/************************************/
struct KEISOKU_TBL init_kojin_keisoku_tbl( void )
{
    static struct KEISOKU_TBL tbl = {
        0, 0, "       ", 0, "       ", 0, "       ", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    };
    return tbl;
}
```

```
/************************************/
/*  nyuukai.c                       */
/*    ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ﾌｧｲﾙ       */
/************************************/
#include <stdio.h>
#include <string.h>

#include "common.h"
#include "main.h"
#include "nyuukai.h"

extern int akicode_tbl[ MEMBER_MAX + 1 ];                   /* ｱｷｺｰﾄﾞ ﾋｮｳ */
extern int codedata_tbl[ MEMBER_MAX ];                      /* ｺｰﾄﾞﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ */
extern struct KEISOKU_TBL kojin_keisoku_tbl;                /* ｺｼﾞﾝﾍﾞﾂ ｹｲｿｸ ﾃﾞｰﾀﾋｮｳ */

/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                  */
/*   ﾒｲﾝ                            */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                    */
/*   ﾘﾀｰﾝ   : 0:OK                  */
/*            -1:NG                 */
/************************************/
int nyuukai_touroku( void )
{
    int    ret;             /* ﾘﾀｰﾝ ｺｰﾄﾞ        */
    int    kaiin_code;      /* ﾄｳﾛｸ ｶｲｲﾝ ｺｰﾄﾞ   */
    long   fptr;            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ       */
    char   msg[ 64 ];       /* ﾒｯｾｰｼﾞ ｴﾘｱ       */

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ READ -> NG ? */
    if( (ret = akicode_tbl_read( )) == NG ) {
        return ret;
    }

    /* ｱｷ ｺｰﾄﾞ ｱﾘ ? */
    if( akicode_tbl[ 0 ] <= 0 ) {
        printf( "¥n ｻﾞﾝﾈﾝﾅｶﾞﾗ ﾀﾀﾞｲﾏ ﾒﾝﾊﾞｰ ﾉ ｱｷｶﾞ ｱﾘﾏｾﾝ¥n" );
        ret = OK;
        return ret;
    }

    /* ｱｷ ｺｰﾄﾞ ｶｸﾆﾝ */
    sprintf( msg, "¥n ｶｲｲﾝ ｺｰﾄﾞ ﾊ %d ﾃﾞｽ. ﾖﾛｼｲ ﾃﾞｽｶ( Y/N )", akicode_tbl[ 1 ] );

    if( (ret = kakunin_input( msg )) == OK ) {
        /* ﾄｳﾛｸ ｶｲｲﾝ ｺｰﾄﾞ ﾀｲﾋ */
        kaiin_code = akicode_tbl[ 1 ];

        /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ｺｳｼﾝ -> OK ? */
        if( (ret = akicode_tbl_update( )) == OK ) {

            /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾂｲｶ -> OK ? */
            if( (ret = keisoku_tbl_add( &fptr, kaiin_code )) == OK ) {

                /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ｺｳｼﾝ */
                ret = codedata_tbl_update( kaiin_code, fptr );
            }
        }
    }

    if( ret == OK ) {
        printf( "¥n ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ ｶﾞ ｼｭｳﾘｮｳ ｼﾏｼﾀ" );
    }

    return ret;
}
```

```
/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                      */
/*   ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ ｺｳｼﾝ               */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                         */
/*   ﾘﾀｰﾝ   : 0:OK                      */
/*           -1:NG                      */
/************************************/
static int akicode_tbl_update( void )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ         */
    int     i;                          /* ｲﾝﾃﾞｯｸｽ           */
    int     cnt;                        /* ｱｷ ｺｰﾄﾞ ｽｳ        */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ        */
    char    *fname = AKICODE_TBL_NAME;/* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ */

    /* ｱｷ ｺｰﾄﾞ ｹﾝｽｳ ｾｯﾄ */
    cnt = akicode_tbl[ 0 ];

    /* ｱｷ ｺｰﾄﾞ ﾍﾝｼｭｳ */
    for( i = 1; i < cnt; i++ ) {
        if( akicode_tbl[ i + 1 ] == 0 ) {
            break;
        }
        akicode_tbl[ i ] = akicode_tbl[ i + 1 ];
    }

    akicode_tbl[ i ] = 0;

    /* ｱｷ ｺｰﾄﾞ ｹﾝｽｳ ｾｯﾄ */
    akicode_tbl[ 0 ] = cnt - 1;

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "w+b" )) == NULL ) {
        printf( "¥n ｱｷｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)akicode_tbl,
              sizeof( int ) * (akicode_tbl[ 0 ] + 1), 1, fp )) != 1 ) {
        printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                      */
/*   ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾂｲｶ              */
/*   ﾊﾟﾗﾒｰﾀ : ｹｲｿｸ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ            */
/*            ﾄｳﾛｸ ｶｲｲﾝ ｺｰﾄﾞ             */
/*   ﾘﾀｰﾝ   : 0:OK                      */
/*           -1:NG                      */
/************************************/
static int keisoku_tbl_add( long *fptr, int kaiin_code )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ           */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ          */
    char    *fname = KEISOKU_TBL_NAME;/* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ */
```

```
    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "a+b" )) == NULL ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ SEEK -> OK ﾃﾞ ﾅｲ ? */
    if( (ret = fseek( fp, 0L, SEEK_END )) != OK ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ SEEK ｴﾗｰ" );
        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
        return NG;
    }

    /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ ｼｭﾄｸ */
    *fptr = ftell( fp );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ｸﾘｱ */
    kojin_keisoku_tbl = init_kojin_keisoku_tbl();

    /* ｶｲｲﾝ ｺｰﾄﾞ ｾｯﾄ */
    kojin_keisoku_tbl.kaiin_code = kaiin_code;

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)&kojin_keisoku_tbl, sizeof( kojin_keisoku_tbl ),
              1, fp )) != 1 ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*  ﾆｭｳｶｲ ﾄｳﾛｸ ｼｮﾘ                   */
/*   ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ｺｳｼﾝ   */
/*   ﾊﾟﾗﾒｰﾀ : ﾄｳﾛｸ ｶｲｲﾝ ｺｰﾄﾞ          */
/*            ｹｲｿｸ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ        */
/*   ﾘﾀｰﾝ   : 0:OK                   */
/*            -1:NG                  */
/************************************/
static int codedata_tbl_update( int kaiin_code, long fptr )
{
    int     ret;                            /* ﾘﾀｰﾝ ｺｰﾄﾞ              */
    FILE    *fp;                            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ              */
    char    *fname = CODEDATA_TBL_NAME;     /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ */

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "r+b" )) == NULL ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ -> 1 ｲｶﾞｲ ? */
    if( (ret = fread( (char *)codedata_tbl, sizeof( codedata_tbl ), 1, fp ) )
             != 1 ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
        ret = NG;
    }
    else {
        /* ｶﾞｲﾄｳ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ ｾｯﾄ */
        codedata_tbl[ kaiin_code - 1 ] =
```

```
            (int)( (fptr / sizeof( struct KEISOKU_TBL )) + 1 );

        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ ヲ ｾﾝﾄｳ ﾆ SEEK -> OK ﾃﾞ ﾅｲ ? */
        if( (ret = fseek( fp, 0L, SEEK_SET )) != OK ) {
            printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ SEEK ｴﾗｰ" );

            /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
            fclose( fp );
            return NG;
        }

        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
        if( (ret = fwrite( (char *)codedata_tbl, sizeof( codedata_tbl ), 1,
                fp )) != 1 ) {
            printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
            ret = NG;
        }
        else {
            ret = OK;
        }
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}
```

```
/************************************/
/*  keisoku.c                       */
/*  ｹｲｿｸｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ﾌｧｲﾙ       */
/************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

#include "common.h"
#include "main.h"
#include "keisoku.h"

extern int akicode_tbl[ MEMBER_MAX + 1 ];                  /* ｱｷｺｰﾄﾞ ﾋｮｳ */
extern int codedata_tbl[ MEMBER_MAX ];                     /* ｺｰﾄﾞﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ */
extern struct KEISOKU_TBL kojin_keisoku_tbl;               /* ｺｼﾞﾝﾍﾞﾂ ｹｲｿｸ ﾃﾞｰﾀﾋｮｳ */
extern struct KEISOKU_TBL sort_keisoku_tbl[ MEMBER_MAX ]; /* ｿｰﾄ ﾖｳ ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ */


/* ﾆｭｳﾘｮｸ ｹｲｿｸ ﾃﾞｰﾀ ﾃｰﾌﾞﾙ */
static struct  KEISOKU_INPUT   keisoku_indata[ 5 ];

/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ             */
/*   ﾒｲﾝ                            */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                    */
/*   ﾘﾀｰﾝ   : 0:OK                  */
/*            -1:NG                 */
/************************************/
int keisoku_input( void )
{
    int     ret;                /* ﾘﾀｰﾝ ｺｰﾄﾞ        */
    int     i;                  /* ｲﾝﾃﾞｯｸｽ          */
    int     kaiin_code;         /* ｹｲｿｸ ｶｲｲﾝ ｺｰﾄﾞ   */
    int     undou_sisuu;        /* ｻﾝｼｭﾂ ｳﾝﾄﾞｳ ｼｽｳ  */
    char    msg[ 64 ];          /* ﾒｯｾｰｼﾞ ｴﾘｱ       */
    char    keisoku_date[ 9 ];  /* ｹｲｿｸ ﾋﾂﾞｹ        */

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ READ -> NG ? */
    if( (ret = akicode_tbl_read( )) == NG ) {
        return ret;
    }

    /* ﾆｭｳｶｲｼｬ ﾁｪｯｸ */
    if( akicode_tbl[ 0 ] >= MEMBER_MAX ) {
        printf( "¥n ｹﾞﾝｻﾞｲ ﾆｭｳｶｲｼｬ ｶﾞ ｲﾏｾﾝ" );
        return NG;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ -> NG ? */
    if( (ret = codedata_tbl_read( )) == NG ) {
        return ret;
    }

    /* ｹｲｿｸ ｶｲｲﾝ ｺｰﾄﾞ ﾆｭｳﾘｮｸ */
    kaiin_code_input( &kaiin_code );

    /* ｹｲｿｸ ﾋﾂﾞｹ ﾆｭｳﾘｮｸ */
    keisoku_date_input( keisoku_date );

    /* ｹｲｿｸﾁ ﾆｭｳﾘｮｸ ｼｮﾘ */
    for( i = 0; i < 5; i++ ) {
        keisoku_data_input( i );
    }

    /* ﾆｭｳﾘｮｸ ﾃﾞｰﾀ ﾋｮｳｼﾞ */
    keisoku_data_disp( kaiin_code, keisoku_date );

    /* ﾆｭｳﾘｮｸ ｶｸﾆﾝ */
    strcpy( msg, "¥n ﾆｭｳﾘｮｸ ﾊ ﾖﾛｼｲ ﾃﾞｽｶ( Y/N )" );
```

```
    /* 'N' OR 'n' ﾆｭｳﾘｮｸ -> OK ﾃﾞ ﾅｲ ? */
    if( (ret = kakunin_input( msg )) != OK ) {
        return NG;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ READ -> NG ? */
    if( (ret = kojin_data_read( kaiin_code )) == NG ) {
        return ret;
    }

    /* ｹｲｿｸﾁ ｹｲｻﾝ ｼｮﾘ */
    undou_sisuu_keisan( &undou_sisuu );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ SET */
    kojin_data_update( kaiin_code, keisoku_date, undou_sisuu );

    /* ｹｲｿｸ ｹｯｶ ﾃﾞｰﾀ ﾋｮｳｼﾞ */
    kojin_data_disp( kaiin_code, "¥n ** ｹｲｿｸ ｹｯｶ ﾃﾞｰﾀ **" );

    /* ｷｰ ﾆｭｳﾘｮｸ ﾏﾁ */
    while( getchar( ) != '¥n' ) ;
    printf( "¥n ﾘﾀｰﾝ ｷｰ ｦ ｵｼﾃｸﾀﾞｻｲ." );
    getchar( );

    /* ｹｲｿｸ ｹｯｶ ｼﾞｭﾝｲ ｿｰﾄ ﾋｮｳｼﾞ */
    ret = keisoku_rank( );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ               */
/*   ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ    */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                      */
/*   ﾘﾀｰﾝ   : 0:OK                    */
/*           -1:NG                    */
/************************************/
static int codedata_tbl_read( void )
{
    int     ret;                            /* ﾘﾀｰﾝ ｺｰﾄﾞ                */
    FILE    *fp;                            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ               */
    char    *fname = CODEDATA_TBL_NAME;     /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ */

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "rb" )) == NULL ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ -> 1 ｲｶﾞｲ ? */
    if( (ret = fread( (char *)codedata_tbl, sizeof( codedata_tbl ), 1, fp ))
            != 1 ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}
```

```
/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｶｲｲﾝｺｰﾄﾞ ﾆｭｳﾘｮｸ                     */
/*   ﾊﾟﾗﾒｰﾀ : ﾆｭｳﾘｮｸ ｶｲｲﾝ ｺｰﾄﾞ           */
/*   ﾘﾀｰﾝ   : ﾅｼ                          */
/************************************/
static void kaiin_code_input( int *kaiin_code )
{
    int     loop = TRUE;     /* ﾙｰﾌﾟ ﾌﾗｸﾞ     */
    char    work[ 128 ];     /* ﾆｭｳﾘｮｸ ﾜｰｸ    */

    while( loop ) {
        printf( "¥n ｹｲｿｸｼｬ ﾉ ｶｲｲﾝ ｺｰﾄﾞ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ" );
        printf( "¥n ? " );

        /* ｶｲｲﾝ ｺｰﾄﾞ ﾆｭｳﾘｮｸ */
        work[ 0 ] = '¥0';
        scanf( "%s", work );

        /* ﾆｭｰﾒﾘｯｸ ﾁｪｯｸ -> ｽｳﾁ ｲｶﾞｲ ? */
        if( strspn( work, "1234567890" ) < strlen( work ) ) {
            printf( "¥n ｽｳﾁ ｲｶﾞｲ ｶﾞ ﾆｭｳﾘｮｸ ｻﾚﾏｼﾀ" );
            continue;
        }

        /* ﾆｭｳﾘｮｸ ﾊﾝｲ ﾁｪｯｸ( 0 < kaiin_code <= MEMBER_MAX ) */
        *kaiin_code = atoi( work );
        if( *kaiin_code > MEMBER_MAX || *kaiin_code <= 0 ) {
            printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
            continue;
        }

        /* ｶｲｲﾝｺｰﾄﾞ ﾄｳﾛｸ ﾁｪｯｸ -> ﾐﾄｳﾛｸ ? */
        if( codedata_tbl[ *kaiin_code - 1 ] == 0 ) {
            printf( "¥n ｺﾉ ｶｲｲﾝｺｰﾄﾞ ﾊ ﾐﾄｳﾛｸ ﾃﾞｽ" );
            continue;
        }
        break;
    }
    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｹｲｿｸ ﾋﾂﾞｹ ﾆｭｳﾘｮｸ                     */
/*   ﾊﾟﾗﾒｰﾀ : ﾆｭｳﾘｮｸ ﾋﾂﾞｹ                 */
/*   ﾘﾀｰﾝ   : ﾅｼ                          */
/************************************/
static void keisoku_date_input( char *keisoku_date )
{
    int     loop = TRUE;     /* ﾙｰﾌﾟ ﾌﾗｸﾞ     */
    int     chk_date;        /* ﾋﾂﾞｹ ｽｳﾁ      */
    char    conv[ 3 ];       /* ｽｳﾁ ﾍﾝｶﾝ ﾖｳ   */
    char    work[ 128 ];     /* ﾆｭｳﾘｮｸ ﾜｰｸ    */

    while( loop ) {
        printf( "¥n ﾋﾂﾞｹ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ( YYYYMMDD )" );
        printf( "¥n ? " );

        /* ﾋﾂﾞｹ ﾆｭｳﾘｮｸ */
        work[ 0 ] = '¥0';
        scanf( "%s", work );

        /* ﾆｭｳﾘｮｸ ｹﾀｽｳ ﾁｪｯｸ -> 8 ｲｶﾞｲ ? */
        if( strlen( work ) != 8 ) {
            printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
            continue;
```

```
        }

        /* ﾆｭｰﾒﾘｯｸ ﾁｪｯｸ -> ｽｳﾁｲｶﾞｲ ? */
        if( strspn( work, "1234567890" ) < strlen( work ) ) {
            printf( "¥n ｽｳﾁ ｲｶﾞｲ ｶﾞ ﾆｭｳﾘｮｸ ｻﾚﾏｼﾀ" );
            continue;
        }

        /* ﾂｷ ﾁｪｯｸ */
        conv[0] = work[4];
        conv[1] = work[5];
        conv[2] = '¥0';

        chk_date = atoi( conv );
        if( chk_date > 12 || chk_date < 1 ) {
            printf( "¥n ﾋﾂﾞｹ( ﾂｷ ) ﾆｭｳﾘｮｸ ｴﾗｰ ﾃﾞｽ" );
            continue;
        }

        /* ﾋ ﾁｪｯｸ */
        conv[0] = work[6];
        conv[1] = work[7];
        conv[2] = '¥0';
        chk_date = atoi( conv );
        if( chk_date > 31 || chk_date < 1 ) {
            printf( "¥n ﾋﾂﾞｹ( ﾋ ) ﾆｭｳﾘｮｸ ｴﾗｰ ﾃﾞｽ" );
            continue;
        }

        break;
    }

    /* ﾆｭｳﾘｮｸ ﾃﾞｰﾀ ｾｯﾄ */
    strcpy(keisoku_date, work);
    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｹｲｿｸﾁ ﾆｭｳﾘｮｸ ｼｮﾘ                   */
/*   ﾊﾟﾗﾒｰﾀ : ｲﾝﾃﾞｯｸｽ                    */
/*   ﾘﾀｰﾝ   : ﾅｼ                         */
/************************************/
static void keisoku_data_input( int idx )
{
    int     i;                  /* ｲﾝﾃﾞｯｸｽ      */
    int     loop = TRUE;        /* ﾙｰﾌﾟ ﾌﾗｸﾞ     */
    char    work[ 3 ][ 128 ];   /* ﾆｭｳﾘｮｸ ﾜｰｸ    */

    while( loop ) {
        printf( "¥n ｳﾝﾄﾞｳ %d ﾉ ｹｲｿｸ ﾃﾞｰﾀ ﾖ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ", idx + 1 );
        printf( "¥n   ﾌｶ(999)  ｾｯﾄ(100)  ｶｲｽｳ(100)" );
        printf( "¥n ? " );

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾆｭｳﾘｮｸ */
        work[ 0 ][ 0 ] = '¥0';
        work[ 1 ][ 0 ] = '¥0';
        work[ 2 ][ 0 ] = '¥0';
        scanf( "%s %s %s", work[ 0 ], work[ 1 ], work[ 2 ] );

        for( i = 0; i < 3; i++ ) {

            /* ﾆｭｳﾘｮｸ ｹﾀｽｳ ﾁｪｯｸ -> 3 ﾖﾘｵｵｷｲ ? */
            if( strlen( work[ i ] ) > 3 ) {
                printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
                break;
            }
```

```
            /* ﾆｭｰﾒﾘｯｸ ﾁｪｯｸ -> ｽｳﾁｲｶﾞｲ ? */
            if( strspn( work[ i ], "1234567890" ) < strlen( work[ i ] ) ) {
                printf( "¥n ｽｳﾁ ｲｶﾞｲ ｶﾞ ﾆｭｳﾘｮｸ ｻﾚﾏｼﾀ" );
                break;
            }

            /* ﾌｶ ｲｶﾞｲ ﾉ ｼﾞｮｳｹﾞﾝ ﾁｪｯｸ -> 100 ﾖﾘｵｵｷｲ ? */
            if( i != 0 ) {
                if( atoi( work[ i ] ) > 100 ) {
                    printf( "¥n ｼﾞｮｳｹﾞﾝ(100) ﾖ ｺｴﾃｲﾏｽ" );
                    break;
                }
            }
        }

        if( i < 3 ) {
            continue;
        }

        break;
    }

    /* ﾆｭｳﾘｮｸ ﾃﾞｰﾀ ﾃｰﾌﾞﾙ ｾｯﾄ */
    keisoku_indata[ idx ].huka = atoi( work[ 0 ] );
    keisoku_indata[ idx ].set = atoi( work[ 1 ] );
    keisoku_indata[ idx ].kaisuu = atoi( work[ 2 ] );

    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ﾆｭｳﾘｮｸ ｹｲｿｸﾁ ﾋｮｳｼﾞ ｼｮﾘ             */
/*   ﾊﾟﾗﾒｰﾀ : ｶｲｲﾝ ｺｰﾄﾞ                   */
/*            ﾋﾂﾞｹ                        */
/*   ﾘﾀｰﾝ   : ﾅｼ                          */
/************************************/
static void keisoku_data_disp( int kaiin_code, char *keisoku_date )
{
    int     i;          /* ｲﾝﾃﾞｯｸｽ      */

    printf( "¥n ** ﾆｭｳﾘｮｸ ｹｲｿｸﾁ ﾃﾞｰﾀ **" );
    printf( "¥n ｶｲｲﾝｺｰﾄﾞ %3d", kaiin_code );
    printf( "¥n ﾋﾂﾞｹ    %4.4s-%2.2s-%2.2s",
        ( keisoku_date + 0 ), ( keisoku_date + 4 ), ( keisoku_date + 6 ) );

    printf( "¥n¥n ｳﾝﾄﾞｳ    ﾌｶ    ｾｯﾄ    ｶｲｽｳ" );

    for( i = 0; i < 5; i++ ) {
        printf( "¥n   %d    %3d    %3d    %3d", i + 1,
            keisoku_indata[ i ].huka,
            keisoku_indata[ i ].set,
            keisoku_indata[ i ].kaisuu );
    }

    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｳﾝﾄﾞｳ ｼｽｳ ｹｲｻﾝ ｼｮﾘ                 */
/*   ﾊﾟﾗﾒｰﾀ : ｳﾝﾄﾞｳ ｼｽｳ                   */
/*   ﾘﾀｰﾝ   : ﾅｼ                          */
/************************************/
static void undou_sisuu_keisan( int *undou_sisuu )
```

```
{
    int     i;              /* ｲﾝﾃﾞｯｸｽ      */
    double  sisuu;          /* ｹｲｻﾝ ﾜｰｸ     */
    double  sisuu_total;    /* ｹｲｻﾝﾁ ｺﾞｳｹｲ  */
    static  double  huka_sisuu[ 5 ] = { 0.24, 0.36, 0.52, 1.05, 2.13 };

    sisuu_total = 0.0;
    for( i = 0; i < 5; i++ ) {

        /* ｶｲｽｳ 1ｲｶ ? */
        if( keisoku_indata[ i ].kaisuu <= 1 ) {
            continue;
        }

        sisuu = huka_sisuu[ i ] * (double)keisoku_indata[ i ].huka *
              (double)keisoku_indata[ i ].set *
              ((pow( (double)keisoku_indata[ i ].kaisuu, 2.0 )) /
              ((double)keisoku_indata[ i ].kaisuu - 1.0));

        /* ｼｸﾞﾏ(ﾙｲｹｲ) ﾉ ｹｲｻﾝ */
        sisuu_total += sisuu;
    }

    /* ﾍｲﾎｳｺﾝ ﾉ ｻﾝｼｭﾂ */
    *undou_sisuu = (int)sqrt( sisuu_total );

    return;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                */
/*   ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ｺｳｼﾝ            */
/*   ﾊﾟﾗﾒｰﾀ : ｶｲｲﾝ ｺｰﾄﾞ                  */
/*            ﾋﾂﾞｹ                      */
/*            ｳﾝﾄﾞｳ ｼｽｳ                  */
/*   ﾘﾀｰﾝ   : 0:OK                       */
/*            -1:NG                     */
/************************************/
static int kojin_data_update( int kaiin_code, char *keisoku_date, int undou_sisuu
 )
{
    int    ret;                         /* ﾘﾀｰﾝ ｺｰﾄﾞ            */
    long   fptr;                        /* ｹｲｿｸ ﾃﾞｰﾀ ﾌｧｲﾙ ﾎﾟｲﾝﾀ */
    FILE   *fp;                         /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ          */
    char   *fname = KEISOKU_TBL_NAME;/* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ   */
    int    i;                           /* ｲﾝﾃﾞｯｸｽ             */

    /* 1ｶｲﾒ ? */
    if( kojin_keisoku_tbl.count <= 0 ) {
        strcpy( kojin_keisoku_tbl.first_date, keisoku_date );
        kojin_keisoku_tbl.first_data = undou_sisuu;

        strcpy( kojin_keisoku_tbl.max_date, keisoku_date );
        kojin_keisoku_tbl.max_data = undou_sisuu;
    }
    else {

        /* ｻｲｺｳ ｷﾛｸ ? */
        if( kojin_keisoku_tbl.max_data < undou_sisuu ) {
            strcpy( kojin_keisoku_tbl.max_date, keisoku_date );
            kojin_keisoku_tbl.max_data = undou_sisuu;
        }
        for (i = sizeof kojin_keisoku_tbl.soku_data / sizeof(int) - 1; i > 0; i-- )
            kojin_keisoku_tbl.soku_data[ i ] = kojin_keisoku_tbl.soku_data[ i - 1 ];
    }

    strcpy( kojin_keisoku_tbl.soku_date, keisoku_date );
    kojin_keisoku_tbl.soku_data[ 0 ] = undou_sisuu;
```

```
    kojin_keisoku_tbl.count++;

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "r+b" )) == NULL ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｶﾞｲﾄｳ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ ｾｯﾄ */
    fptr = ( codedata_tbl[ kaiin_code - 1 ] - 1 ) *
        sizeof( struct KEISOKU_TBL );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ SEEK -> OK ﾃﾞ ﾅｲ ? */
    if( (ret = fseek( fp, fptr, SEEK_SET )) != OK ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ SEEK ｴﾗｰ" );

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
        return NG;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)&kojin_keisoku_tbl, sizeof( kojin_keisoku_tbl ),
              1, fp )) != 1 ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｹｲｿｸ ｹｯｶ ｼﾞｭﾝｲ ｿｰﾄ ﾋｮｳｼﾞ            */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                         */
/*   ﾘﾀｰﾝ   : 0:OK                       */
/*            -1:NG                      */
/************************************/
static int keisoku_rank( void )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ            */
    int     i;                          /* ｲﾝﾃﾞｯｸｽ              */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ           */
    char    *fname = KEISOKU_TBL_NAME;/* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ   */

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "rb" )) == NULL ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    i = 0;
    for( ; ; ) {

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ -> 1 ｲｶﾞｲ ? */
        if( (ret = fread( (char *)&sort_keisoku_tbl[ i ],
                  sizeof( struct KEISOKU_TBL ), 1, fp )) != 1 ) {

            /* READ ｴﾗｰ ? */
            if( ferror( fp ) != 0 ) {
                printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                ret = NG;
```

```
            }
            else {
                /* ﾌｧｲﾙ EOF ﾃﾞ ﾅｲ ? */
                if( feof( fp ) == 0 ) {
                    printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                    ret = NG;
                }
                else {
                    ret = OK;
                }
            }
        }

        /* READ ｴﾗｰ ? */
        if( ret == NG ) {
            break;
        }

        /* ｹｲｿｸ ﾃﾞｰﾀ ｱﾘ ? */
        if( sort_keisoku_tbl[ i ].count != 0 ) {
            i++;
        }

        /* eof ? */
        if( ret == OK ) {
            break;
        }
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    if( ret == OK ) {
        /* ｹｲｿｸ ﾃﾞｰﾀ ｿｰﾄ */
        keisoku_data_sort( i );
        /* ｹｲｿｸ ﾃﾞｰﾀ ｿｰﾄ ｹｯｶ ﾋｮｳｼﾞ */
        keisoku_sort_disp( i );
    }

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｹｲｿｸ ｹｯｶ ｿｰﾄ                      */
/*   ﾊﾟﾗﾒｰﾀ : ｹｲｿｸ ﾃﾞｰﾀ ｹﾝｽｳ            */
/*   ﾘﾀｰﾝ   : ﾅｼ                        */
/************************************/
static void keisoku_data_sort( int cnt )
{
    int     i;                      /* ｲﾝﾃﾞｯｸｽ      */
    int     j;                      /* ｲﾝﾃﾞｯｸｽ      */
    struct  KEISOKU_TBL  work;      /* ｽﾜｯﾌﾟ ﾖｳ ｴﾘｱ */

    /* ﾃﾞｰﾀ ｿｰﾄ */
    for( i = 0; i < cnt - 1; i++ ) {
        for( j = i + 1; j < cnt; j++ ) {
            if( sort_keisoku_tbl[ i ].max_data <
                sort_keisoku_tbl[ j ].max_data ) {

                work = sort_keisoku_tbl[ i ];
                sort_keisoku_tbl[ i ] = sort_keisoku_tbl[ j ];
                sort_keisoku_tbl[ j ] = work;
            }
        }
    }
    return;
}
```

```
/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ｹｲｿｸ ｷﾛｸ ﾆｭｳﾘｮｸ ｼｮﾘ                 */
/*   ｹｲｿｸ ｹｯｶ ﾋｮｳｼﾞ                     */
/*   ﾊﾟﾗﾒｰﾀ : ｹｲｿｸ ﾃﾞｰﾀ ｹﾝｽｳ             */
/*   ﾘﾀｰﾝ   : ﾅｼ                          */
/************************************/
static void keisoku_sort_disp( int cnt )
{
    int     i;        /* ｲﾝﾃﾞｯｸｽ  */

    printf( "¥n¥n ** ｼﾞｭﾝｲ ﾋｮｳ **" );
    printf( "¥n ｼﾞｭﾝｲ ｶｲｲﾝｺｰﾄﾞ  ｻｲｺｳﾃﾞｰﾀ  ﾋﾂﾞｹ" );

    for( i = 0; i < cnt; i++ ) {

        /* 10ｲ ﾏﾃﾞ ﾋｮｳｼﾞ ｽﾙ */
        if( i >= 10 ) {
            break;
        }

        printf( "¥n   %2d   %3d       %4d   %4.4s-%2.2s-%2.2s", i + 1,
            sort_keisoku_tbl[ i ].kaiin_code,
            sort_keisoku_tbl[ i ].max_data,
            &sort_keisoku_tbl[ i ].max_date[ 0 ],
            &sort_keisoku_tbl[ i ].max_date[ 4 ],
            &sort_keisoku_tbl[ i ].max_date[ 6 ] );
    }
    return;
}
```

```
/************************************/
/*  sakujyo.c                       */
/*    ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ﾌｧｲﾙ       */
/************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#include "common.h"
#include "main.h"
#include "sakujyo.h"

extern int akicode_tbl[ MEMBER_MAX + 1 ];                   /* ｱｷｺｰﾄﾞ ﾋｮｳ */
extern int codedata_tbl[ MEMBER_MAX ];                      /* ｺｰﾄﾞﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ */
extern struct KEISOKU_TBL kojin_keisoku_tbl;                /* ｺｼﾞﾝﾍﾞｯ ｹｲｿｸ ﾃﾞｰﾀﾋｮｳ */

/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                 */
/*  ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ                  */
/*   ﾒｲﾝ                            */
/*   ﾊﾟﾗﾒｰﾀ : ﾅｼ                    */
/*   ﾘﾀｰﾝ   : 0:OK                  */
/*           -1:NG                  */
/************************************/
int touroku_sakujyo( void )
{
    int     ret;            /* ﾘﾀｰﾝ ｺｰﾄﾞ        */
    int     loop = TRUE;    /* ﾙｰﾌﾟ ﾌﾗｸﾞ        */
    int     kaiin_code;     /* ｻｸｼﾞｮ ｶｲｲﾝ ｺｰﾄﾞ  */
    char    work[ 128 ];    /* ﾆｭｳﾘｮｸ ﾜｰｸ       */

    while( loop ) {
        printf( "¥n ﾀｲｶｲｼｬ ﾉ ｶｲｲﾝ ｺｰﾄﾞ ｦ ﾆｭｳﾘｮｸ ｼﾃｸﾀﾞｻｲ" );
        printf( "¥n ? " );

        /* ｶｲｲﾝ ｺｰﾄﾞ ﾆｭｳﾘｮｸ */
        work[ 0 ] = '¥0';
        scanf( "%s", work );

        /* ﾆｭｰﾒﾘｯｸ ﾁｪｯｸ -> ｽｳﾁｲｶﾞｲ ? */
        if( strspn( work, "1234567890" ) < strlen( work ) ) {
            printf( "¥n ｽｳﾁ ｲｶﾞｲ ｶﾞ ﾆｭｳﾘｮｸ ｻﾚﾏｼﾀ" );
            continue;
        }

        /* ﾆｭｰﾘｮｸ ﾊﾝｲ ﾁｪｯｸ -> ( 0 < kaiin_code <= MEMBER_MAX ) ? */
        kaiin_code = atoi( work );
        if( kaiin_code > MEMBER_MAX || kaiin_code <= 0 ) {
            printf( "¥n ﾆｭｳﾘｮｸ ﾐｽ ﾃﾞｽ" );
            continue;
        }

        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ｻｸｼﾞｮ -> OK ? */
        if( (ret = codedata_tbl_delete( kaiin_code )) == OK ) {
            /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾂｲｶ -> OK ? */
            if( (ret = akicode_tbl_add( kaiin_code )) == OK ) {
                /* ﾒｲﾝ ﾆ ﾓﾄﾞﾙ */
                loop = FALSE;
            }
        }
        else {
            /* ﾒｲﾝ ﾆ ﾓﾄﾞﾙ */
            loop = FALSE;
        }
    }

    if( ret == OK ) {
        printf( "¥n ﾆｭｳｶｲ ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ ｶﾞ ｼｭｳﾘｮｳ ｼﾏｼﾀ" );
    }

    return ret;
}
```

```
/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ                     */
/*   ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ｺｳｼﾝ      */
/*   ﾊﾟﾗﾒｰﾀ : ｻｸｼﾞｮ ｶｲｲﾝ ｺｰﾄﾞ           */
/*   ﾘﾀｰﾝ   : 0:OK                     */
/*            1:CANCEL                 */
/*           -1:NG                     */
/************************************/
static int codedata_tbl_delete( int kaiin_code )
{
    int     ret;                            /* ﾘﾀｰﾝ ｺｰﾄﾞ                */
    int     i;                              /* ｲﾝﾃﾞｯｸｽ                   */
    char    msg[ 64 ];                      /* ﾒｯｾｰｼﾞ ｴﾘｱ                */
    FILE    *fp;                            /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ                */
    char    *fname = CODEDATA_TBL_NAME;     /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ */

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "r+b" )) == NULL ) {
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ -> 1 ｲｶﾞｲ ? */
    if( (ret = fread( (char *)codedata_tbl, sizeof( codedata_tbl ), 1, fp ) )
             != 1 ) {
        /* READ ｴﾗｰ */
        printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );

        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
        return NG;
    }

    /* ｶｲｲﾝ ﾄｳﾛｸ ﾁｪｯｸ -> ﾐﾄｳﾛｸ ? */
    if( codedata_tbl[ kaiin_code - 1 ] == 0 ) {
        printf( "¥n ｺﾉ ｶｲｲﾝ ｺｰﾄﾞ ﾊ ﾐﾄｳﾛｸ ﾃﾞｽ" );

        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
        return CANCEL;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ READ -> NG ? */
    if( (ret = kojin_data_read( kaiin_code )) == NG ) {
        /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
        fclose( fp );
        return ret;
    }

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳｼﾞ */
    kojin_data_disp( kaiin_code, "¥n ** ｻｸｼﾞｮ ﾃﾞｰﾀ **" );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ｻｸｼﾞｮ ｶｸﾆﾝ */
    sprintf( msg, "¥n¥n ｳｴ ﾉ ﾃﾞｰﾀ ﾖ ｻｸｼﾞｮ ｼﾏｽ. ﾖﾛｼｲ ﾃﾞｽｶ( Y/N )" );

    if( (ret = kakunin_input( msg )) == OK ) {

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ｻｸｼﾞｮ -> OK ? */
        if( (ret = kojin_data_delete( kaiin_code )) == OK ) {

            /* ｹｲｿｸ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ ｺｳｼﾝ */
            for( i = 0; i < MEMBER_MAX; i++ ) {
                if( codedata_tbl[ i ] > codedata_tbl[ kaiin_code - 1 ] ) {
                    codedata_tbl[ i ]--;
                }
            }
            /* ｹｲｿｸ ﾃﾞｰﾀ ﾎﾟｲﾝﾀ ｸﾘｱ */
            codedata_tbl[ kaiin_code - 1 ] = 0;

            /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ ﾉ ｾﾝﾄｳｲﾁ ﾆ SEEK -> OK ? */
```

```
            if( (ret = fseek( fp, 0L, SEEK_SET )) == OK ) {

                /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1ｲｶﾞｲ ? */
                if( (ret = fwrite( (char *)codedata_tbl,
                          sizeof( codedata_tbl ),1, fp )) != 1 ) {
                    printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
                    ret = NG;
                }
                else {
                    ret = OK;
                }
            }
            else {
                printf( "¥n ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ SEEK ｴﾗｰ" );
                ret = NG;
            }
        }
    }
    else {
        /* ｼｮﾘ ﾁｭｳｼ */
        ret = CANCEL;
    }

    /* ｺｰﾄﾞ ﾃﾞｰﾀ ﾀｲｼｮｳ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}



/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ                     */
/*   ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ｻｸｼﾞｮ               */
/*   ﾊﾟﾗﾒｰﾀ : ｻｸｼﾞｮ ｶｲｲﾝ ｺｰﾄﾞ           */
/*   ﾘﾀｰﾝ   : 0:OK                     */
/*           -1:NG                     */
/************************************/
static int kojin_data_delete( int kaiin_code )
{
    int    ret;                         /* ﾘﾀｰﾝ ｺｰﾄﾞ                */
    int    i;                           /* ｲﾝﾃﾞｯｸｽ                   */
    FILE   *fp;                         /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾎﾟｲﾝﾀ */
    FILE   *tmp;                        /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ ﾎﾟｲﾝﾀ        */
    char   *fname = KEISOKU_TBL_NAME; /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ       */
    char   *tmpfl = "keisoku.tmp";    /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ              */

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "rb" )) == NULL ) {
        printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (tmp = fopen( tmpfl, "w+b" )) == NULL ) {
        printf( "¥n ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        fclose( fp );
        return NG;
    }

    i = 0;
    for( ; ; ) {

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ -> 1ｲｶﾞｲ ? */
        if( (ret = fread( (char *)&kojin_keisoku_tbl,
                  sizeof( kojin_keisoku_tbl ), 1, fp )) != 1 ) {
            /* READ ｴﾗｰ ? */
            if( ferror( fp ) != 0 ) {
                printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                ret = NG;
```

```
            }
            else {
                /* ﾌｧｲﾙ EOF ﾃﾞ ﾅｲ ? */
                if( feof( fp ) == 0 ) {
                    printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ READ ｴﾗｰ" );
                    ret = NG;
                }
                else {
                   ret = OK;
                }
            }
            break;
        }

        /* ｻｸｼﾞｮ ﾃﾞｰﾀ ? */
        if( kaiin_code == kojin_keisoku_tbl.kaiin_code ) {
            continue;
        }

        /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
        if( (ret = fwrite( (char *)&kojin_keisoku_tbl,
                  sizeof( kojin_keisoku_tbl ), 1, tmp )) != 1 ) {
            printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
            ret = NG;
            break;
        }
        i++;
    }

    /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ CLOSE */
    fclose( tmp );

    /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    /* ｾｲｼﾞｮｳ ｼｭｳﾘｮｳ ? */
    if( ret == OK ) {

        /* ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ DELETE -> 0 ｲｶﾞｲ ? */
        if( (ret = remove( fname )) != 0 ) {
            printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ｻｸｼﾞｮ ｴﾗｰ" );
            ret = NG;
        }
        else {
            /* ｶｷｺﾐ ﾃﾞｰﾀ ｱﾘ ? */
            if( i > 0 ) {

                /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ ﾖ ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾆ ﾘﾈｰﾑ ｽﾙ -> 0 ｲｶﾞｲ ? */
                if( (ret = rename( tmpfl, fname )) != 0 ) {
                    printf( "¥n ｹｲｿｸ ﾃﾞｰﾀ ﾋｮｳ ﾌｧｲﾙ ﾘﾈｰﾑ ｴﾗｰ" );
                    ret = NG;
                }
            }
            else {
                /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ ｻｸｼﾞｮ */
                remove( tmpfl );
            }
        }
    }
    else {
        /* ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ ｻｸｼﾞｮ */
        remove( tmpfl );
    }

    return ret;
}
```

```
/************************************/
/* ﾒﾝﾊﾞｰ ｶﾝﾘ ﾌﾟﾛｸﾞﾗﾑ                    */
/*  ﾄｳﾛｸ ｻｸｼﾞｮ ｼｮﾘ                     */
/*   ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ ﾂｲｶ               */
/*   ﾊﾟﾗﾒｰﾀ : ｻｸｼﾞｮ ｶｲｲﾝ ｺｰﾄﾞ           */
/*   ﾘﾀｰﾝ   : 0:OK                     */
/*           -1:NG                     */
/************************************/
static int akicode_tbl_add( int kaiin_code )
{
    int     ret;                        /* ﾘﾀｰﾝ ｺｰﾄﾞ         */
    int     cnt;                        /* ｱｷ ｺｰﾄﾞｹﾝｽｳ       */
    FILE    *fp;                        /* ﾌｧｲﾙ ﾎﾟｲﾝﾀ        */
    char    *fname = AKICODE_TBL_NAME;/* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ */

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ READ -> NG ? */
    if( (ret = akicode_tbl_read( )) == NG ) {
        return ret;
    }

    /* ｱｷ ｺｰﾄﾞ ｹﾝｽｳ ｾｯﾄ */
    cnt = akicode_tbl[ 0 ];

    /* ｱｷ ｺｰﾄﾞ ﾃｰﾌﾞﾙ ｾｯﾄ */
    akicode_tbl[ cnt + 1 ] = kaiin_code;

    /* ｱｷ ｺｰﾄﾞ ｹﾝｽｳ ｾｯﾄ */
    akicode_tbl[ 0 ] = cnt + 1;

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ OPEN -> NULL ? */
    if( (fp = fopen( fname, "w+b" )) == NULL ) {
        printf( "¥n ｱｷ ｺｰﾄﾞﾋｮｳ ﾌｧｲﾙ OPEN ｴﾗｰ" );
        return NG;
    }

    /* ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ WRITE -> 1 ｲｶﾞｲ ? */
    if( (ret = fwrite( (char *)akicode_tbl,
              sizeof( int ) * ( akicode_tbl[ 0 ] + 1 ), 1, fp )) != 1 ) {
        printf( "¥n ｱｷ ｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ WRITE ｴﾗｰ" );
        ret = NG;
    }
    else {
        ret = OK;
    }

    /* ｱｷｺｰﾄﾞ ﾋｮｳ ﾌｧｲﾙ CLOSE */
    fclose( fp );

    return ret;
}
```

平成22年 2 月 1 日　第 4 版