2006 年 1 月 18 日 (木) 実施

前回　演習 1　解答例

```
/* ex11-1.c */
#include <stdio.h>
#include <stdlib.h>

#define MAX 256

int main(int argc,char *argv[])
{
    char str[MAX];
    FILE *fin, *fout;

    if (argc != 3)
    {
        printf("利用法：ex11-1 コピー元ファイル名 コピー先ファイル名¥n");
        exit(1);
    }

    if (NULL == (fin = fopen(argv[1],"r")))
    {
        printf("コピー元ファイル%s が開けません。¥n", argv[1]);
        exit(1);
    }

    if (NULL == (fout = fopen(argv[2],"w")))
    {
        printf("コピー先ファイル%s が開けません。¥n", argv[2]);
        exit(1);
    }

    if (fscanf(fin, "%s", str))
    {
        printf("ファイル%s から読み込まれた氏名は%s です。¥n", argv[1], str);
        fprintf(fout,"%s¥n", str);
    }
    else
    {
        fprintf(stderr, "ファイル%s からの氏名の読み込みエラーです。¥n", argv[1]);
        exit(1);
    }

    fclose(fin);
    fclose(fout);

    return 0;
}
```

構造体

レコードと構造体

前回の教材で触れたように，複数の項目に渡るデータを一まとめにしたものを**レコード**という。

例えば，次のように学籍番号，氏名，履修科目コード，点数，評価といった項目による 1 人分のデータを一まとめにしたものは 1 件分のレコードである。

学籍番号	氏名	履修科目コード	点数	評価
0610123	山本勘助	x0621234	100	S

C 言語では，このようなレコードのデータ構造を**構造体**（structure）によって**カプセル化**して表現する。構造体を用いれば，個々の項目を別々の変数で表す場合に比べてデータ間の関係が把握しやすく，プログラムの可読性が増す。

また，C 言語の関数では通常の変数を return 文に書いた場合，1 つのデータしか戻せないが，構造体を return 文に記述することが可能であるので，複数のデータを一まとめにして戻すことができる。

なお，配列同士は要素数が等しくても代入は行えないが，構造体同士はデータ構造が等しければ代入を行うことができる。

構造体の利用

C 言語で構造体を用いる際の一連の流れは次のようになる。

1) レコード設計を行い，**構造体の型定義**を行う　（レコードの各項目をメンバと呼ぶ）
2) **構造体変数の宣言**を行う　（その際に初期化を行うことも可能）
3) 構造体変数のメンバへの代入，メンバの参照

上述の学生データを例にとって，この一連の流れを表すと，次のようになる。

例）
```
struct STUDENT
{
    char         id[8];
    char      name[21];
    char      class[9];
    unsigned int point;
    char          eval;
};

struct STUDENT x;

strcpy(x.id,    "0610123");
strcpy(x.name,  "山本勘助");
strcpy(x.class, "x0621234");
x.point = 100;
x.eval  = 'S';
```

この例で，STUDENT は構造体タグと呼ばれる。ここに書いた形式のほかに，構造体の型定義と構造体変数の宣言とを一度に行うことも可能である。(struct 構造体タグ名{・・・}構造体変数名;)

例題 1

次のプログラムを入力し，翻訳・編集して実行形式のファイルを作成し，実行せよ。ここで，ソースプログラム名は prog12-1.c とする。

```
/* prog12-1.c */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX 5

int main(void)
{
    struct STUDENT
    {
        char         id[8];
        char      name[21];
        char      class[9];
        unsigned int point;
        char          eval;
    };

    struct STUDENT prog[MAX];

    int i;
    char temp[9];
    FILE *fpr;

    if (NULL == (fpr = fopen("stprog.csv","a")))
    {
        printf("stprog データファイルが開けません。¥n");
        exit(1);
    }

    fprintf(fpr,"学籍番号,氏名,履修科目コード,点数,評価¥n");

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

        printf("%d 人目の履修科目コードを入力してください：", i+1);
        scanf(" %s", temp);

        if (strcmp(temp,"x0643234") == 0)
        {
            strcpy(prog[i].class, temp);

            printf("%d 人目の学籍番号を入力してください：", i+1);
            scanf(" %s", prog[i].id);
            printf("%d 人目の氏名を入力してください：", i+1);
            scanf(" %s", prog[i].name);
            printf("%d 人目の点数を入力してください：", i+1);
            scanf(" %u", &prog[i].point);
            printf("%d 人目の評価を入力してください：", i+1);
            scanf(" %c", &prog[i].eval);

            fprintf(fpr,"%s,%s,%s,%u,%c¥n",
              prog[i].id,prog[i].name,prog[i].class,prog[i].point,prog[i].eval);
        }
        else
        {
            printf("履修科目コードが正しくありません。"
```

```
            "もう一度入力してください。¥n");
            i--;
        }
    }

    fclose(fpr);

    return 0;
}
```

【解説】

1. string.h は，文字列操作関数を利用するために必要なヘッダである。(第 6 回教材参照)
2. unsigned int は負号なし整数のデータ型であり，対応する変換指定には%u を用いる。
3. prog は，構造体の配列として宣言されている。この場合，配列の各要素が構造体となる。
4. strcmp は，2 つの引数に書かれた文字列を比較し，等しい場合には 0 を戻す文字列操作のライブラリ関数である。
5. strcpy は，第 2 引数に書かれた文字列を第 1 引数に複写する文字列操作のライブラリ関数である。
6. printf では，複数の文字列リテラルを並べると文字列は繋がって表示される。ここでは，文字列が長いので行を分割するためにこのような書き方をしている。
7. 文字コードが一致せずに再入力を促す場合，既にループ変数 i の値は 1 だけ進んでいるので，i--で元に戻してから，再入力させる。

例題 2

次のプログラムを入力し，翻訳・編集して実行形式のファイルを作成し，実行せよ。ここで，ソースプログラム名は prog12-2.c とする。

```
/* prog12-2.c */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX 5
#define MAXBUF 256

char headln[MAXBUF];

struct STUDENT
{
    char         id[8];
    char      name[21];
    char      class[9];
    char      point[4];
    char       eval[2];
};

void dispdata(struct STUDENT [], int);

int main(void)
```

```
{

    struct STUDENT prog[MAX];

    int i;
    char temp[MAXBUF];
    FILE *fpr;

    if (NULL == (fpr = fopen("stprog.csv","r")))
    {
        printf("stprog データファイルが開けません。¥n");
        exit(1);
    }

    fgets(headln, sizeof headln, fpr);

    for (i=0; i<MAX; i++)
    {
        if (fgets(temp, sizeof temp, fpr) == NULL) break;

        strcpy(prog[i].id, strtok(temp,",¥n"));
        strcpy(prog[i].name, strtok(NULL,",¥n"));
        strcpy(prog[i].class, strtok(NULL,",¥n"));
        strcpy(prog[i].point, strtok(NULL,",¥n"));
        strcpy(prog[i].eval, strtok(NULL,",¥n"));
    }

    dispdata(prog, MAX);

    fclose(fpr);

    return 0;
}

void dispdata(struct STUDENT data[], int n)
{
    int i;

    printf("   %s", headln);

    for (i=0; i<n; i++)
        printf("%d) %s,%s,%s,%s,%s¥n", i+1,
          data[i].id,data[i].name,data[i].class,data[i].point,data[i].eval);
}
```

【解説】

1. fgets は，第 3 引数で指定した入力ストリームから第 2 引数で指定したバイト数分の文字列を読み込み，第 1 引数で指定した配列に格納するライブラリ関数である。
2. strtok は，第 1 引数で指定した文字列を第 2 引数で指定した文字列中のいずれかの文字で区切られる字句（token）の列に分割する。なお，2 回目以降の検索には NULL（空ポインタ）を指定する。
3. ここでは，strcpy を用いて各メンバに字句を複写していく目的で全てのメンバを char 型配列としている。なお，prog[i].point は atoi(prog[i].point) によって数値化することができる。

演習 1

例題 1 のプログラムでは履修科目コードが固定されている。これを第 1 プログラム引数として任意のコードが与えられるようにするプログラムを作成せよ。なお，プログラム中では履修科目コードのチェックは行わないものとする。ここで，ソースプログラム名は ex12-1.c とする。

演習 2

演習 1 のプログラムを改良して，履修科目コード.csv（例えば x0643234.csv）という名前のファイルを開き，データを書き出すプログラムを作成せよ。ここで，ソースプログラム名は ex12-2.c とする。