The Online Grade Book Project

Date: _____

Sample Code to Generate Test Data

Overview

This program was used to generate data for the Departments.ibcs file:

```
//File: GenData
//Purpose : populate some of the Online Grade Book Tables
//  Start from district 10
//  a) Add some school file records
//  b) Add some department file records
//  c) add some course file records

#include <stdio.h>
#include <string.h>
#include <stdlib.h> // for exit
#include "mystrings.h"
#include "parser.h"
#include "IBCS_Ftypes.h"
#include "olGradeBook.h"
#include "olgb.h"
#include "ibcsprocs.h"

// all data will be for district DID = 10
#define  DID 10

int main(int argc, char *argv[]) {
// declare structures and file variables
    int         i,j,k,l,m,rec,n,rc, ret = 0;
    int         KEY = -1;
    int         CONTACT = -1;
    int         ID = -1;
    int         debug=10;
    District    my_dist;
    School      my_school;

    FTOC        school_ftoc;
    FTOC        district_ftoc;
    FTOC        department_ftoc;
    FTOC        course_ftoc;
    FILE        *DFP; // district file pointer
    FILE        *SFP; // School file pointer
    FILE        *CFP; // Course file pointer
    FILE        *DepFP; // department file pointer
    User        my_user;
    Contact     my_contact;
    Course      my_course;
    Department  my_department;
    FileNames   fns;
```

The Online Grade Book Project

```
// 1: get the list of file names
    if (get_fileNames(&fns) != 1) {
              printf("error getting the file names\n");
              exit(1);
          }

// 2: open all files for updating
    DFP   = fopen(fns.districts_fn,"r+");
    SFP   = fopen(fns.schools_fn,"r+");
    CFP   = fopen(fns.courses_fn,"r+");
    DepFP = fopen(fns.departments_fn,"r+");
    if ((DFP == NULL) || (SFP  == NULL) ||
        (CFP == NULL) || (DepFP== NULL)) {
              printf("Error openning files\n");
              exit(1);
              }

 // 3: Read the FTOCs
    i = read_ftoc(SFP,   &school_ftoc);
    j = read_ftoc(DFP,   &district_ftoc);
    k = read_ftoc(DepFP, &department_ftoc);
    l = read_ftoc(CFP,   &course_ftoc);
    if ((i + j + k + l) != 4) {
          printf("Error reading TOCs\n");
          exit(1);
          }


// 4: process the command line arguments, if any
 for (i = 1; i< argc; i++) {
     if (debug > 2) {
        printf("Process command line arg #%d: %s\n",i, argv[i]);
        }
```

```
if (strcmp(argv[i],"-Departments") == 0 ) {
   //Department  my_department;
   //FTOC        department_ftoc;
   //FILE        *DepFP; // department file pointer

    printf("Creating 7 school records\n");
    char *departments[7]= {"GROUP 1: Studies in Languages and Literature",
                           "GROUP 2: Language Acquisition",
                           "GROUP 3 : Individuals and Societies",
                           "GROUP 4: Experimental Sciences",
                           "GROUP 5: Mathematics",
                           "GROUP 6: The Arts",
                           "Advisory"};

   department_ftoc.recsize = sizeof(my_department); // mighta changed
   department_ftoc.recs = 7;
   department_ftoc.nextRec = 8;

   // save the updated FTOC
   ret = fseek(DepFP, 0, SEEK_SET);
   ret = fwrite(&department_ftoc, sizeof(department_ftoc), 1, DepFP);
   // Keep the file pointer here rather than writing to the end of the
   // file. This will rewrite the file each time this program is run

   my_department.sid = 4;
   my_department.distid = DID; // 10;
   my_department.tid = 0; // none assigned
   strcpy(my_department.chair, "To be specified");

   for (i = 0; i < 7; i++) {
       strcpy(my_department.department_name,departments[i]);
       my_department.did = i+1; // 1 to 7
       ret = fwrite(&my_department, sizeof(my_department), 1, DepFP);
       }
// This was included as a reference to make sure all fields get initialized:
//   typedef struct {
//    int  sid;     // school id - unique accross all schools
//    int  did;     // department id
//    int  distid;  // make things easier to find
//    char department_name[100];
//    char chair[100]; // name here if tid = 0
//    int  tid;  // if the chair is a teacher, what is the tid
// } Department;


     } // process Departments
   } // for loop through the arguments
//-------------------------------------------------------------
// done generating data
//-------------------------------------------------------------
   // last step close files
   fclose(DFP);   fclose(SFP);
   fclose(CFP);   fclose(DepFP);
}
```