§7.用户自定义数据类型 - PART VI

补充：

16、 学生作业首行检查程序，检查源程序文件的首行的学号/姓名/班级是否存在及是否匹配

【目录结构说明：】

所有相关的信息均存放在 source 目录及它的子目录下面，source 目录放在可执行文件(命令行方式运行时)/源程序文件(集成环境运行时)的下面，具体结构为：

- ./source/课号-学号：存放学生作业的子目录，同时存在若干个（例：100384-1234567 子目录中存放了选修了 100384 这门课的学号为 1234567 的同学的全部作业）
  - 目录下面可能会有若干*.h/*.c/*.cpp 及其他后缀的文件(*.pdf/*.doc 等)
  - 可能会出现一个文件都没有空目录（例：某位同学登录过，但所有作业都未交）
  - 可能会出现文件不全的目录（例：某位同学未交 3-b1.cpp）
  - 可能会出现某位选课同学的对应目录不存在的情况（例：某同学从未登录过系统）
- ./source/课号.dat ：学生名单文件，同时存在若干个（例：100384.dat 里面存放 100384 这门课的学生名单）
  - 课号.dat 文件中存放该课号所对应的学生名单（无序），格式为每个学生一行，每行的第 1 列为学号，第 2 列为姓名，第 3/4 列为班级，列之间以若干空格/tab 键分隔
  - 课号.dat 文件中出现的学生信息和学生目录可能会不匹配，既有可能是课号.dat 中出现的同学没有对应目录（例：某位同学从未登录过系统），也有可能出现课号.dat 中不存在的同学存在着对应目录（例：某位同学第一周选课并交了作业，然后退课）
- ./source/***.cpp：存放用于替换学生作业的源程序文件中的 main 函数的新 main 函数的文件（例：6-b6-m.cpp）

【检查要求说明：】

- 要检查的首行信息包括学号、姓名、班级三项；其中学号、姓名必须完全匹配，而班级只要部分匹配即可（例：100384.dat 中的“一班”可对应一班/计一班/计算机一班/电子一班等，“1 班”可对应 1 班/计 1 班/计算机 1 班/电子 1 班等）
- 每个被检查的文件(例：3-b1.cpp)只检查首行信息，但如果之前有空行或者只包含空格/tab 键的行，要忽略这些空行并取到首行信息，不能算错误
- 首行信息的注释方式/* */和//两种形式均可，如果首行信息不是注释行则给出“未取到首行信息”的错误
- 首行的三项信息排列顺序任意，但是信息自身不允许含空格（例：名字不能是“张 三”），三个信息之间必须有空格或 tab 键分隔，否则认为是一项信息（例：//张三 1 班 777741，中间无空格/tab 键分隔，则只算一项信息）
- 三项信息与注释字符间有/无空格/tab 键均认为是正确的（例：/* 1234567 张三 一班 */或 /*1234567 张三 一班*/均正确）
- 如果从首行中取到的信息不满三个则给出“首行信息不全”的错误
- 从首行取到三项信息后，进行匹配性检查，具体的规则为：
  - 以文件所在目录名中的学号为基准学号（例：现在正在检查的文件是 100384-1234567 目录下 3-b1.cpp，则基准学号是 1234567）
  - 基准学号/首行信息中的学号必须相同，否则给出“学号不匹配”的错误

- 在学生名单(例：100384.dat)中查找与基准学号匹配的行（如果名单中有两行及以上的学号相同，则取顺序在前者）所对应的姓名和班级，再和首行信息中的姓名/班级去匹配，不匹配则分别给出“姓名不匹配”、“班级不匹配”的错误

- 首行信息如果超过三项，则只取前三项进行检查，后面忽略，不报错（例：//7771234 张三 一班 7774321 李四，则按“7771234 张三 一班”检查，如果这个是张三对应目录中的文件，则检查通过，不会给出“学号不匹配”、“姓名不匹配”等错误）

【程序运行方式:】

通过 main 函数带参数的方式带入多个参数来完成多种功能，具体如下：

执行方式	功能
7-b16 -checktitle 100384 3-b1.cpp	检查课号为 100384 的所有学生的 3-b1.cpp 的首行 1、-checktitle 后面的第 1 参数是课号，第 2 参数是要检查的源程序文件名，课号最大长度为 15 位，源程序文件名最大长度为 31 2、源程序文件名的后缀只能是.cpp/.c/.h 三种，若出现其他后缀或者没有后缀（例：3-b1）的情况，则直接给出“不是源程序文件”的错误并结束即可 3、若某同学目录下无文件则提示“3-b1.cpp 未提交” 4、如果某位同学的目录下有指定文件（例：3-b1.cpp），但是名单中没有该同学(先选课后退课的情况)，则不检查文件也不给出任何错误提示信息
7-b16 -checktitle 100384 all	检查课号为 100384 的所有学生的所有源程序文件 1、只有小写的 all 代表全部文件，其余都是指某个具体文件名（例：参数是 ALL，则给出“不是源程序文件”的错误并结束即可） 2、检查某位同学对应目录下所有的*.cpp/*.c/*.h 文件的首行信息，每个文件都要给出正确或错误的信息，其它后缀的文件不需要检查也不给出提示信息（例：某目录下有 3-5.pdf 则不检查也不给出任何信息） 3、如果某位同学的对应目录存在，但是名单中没有该同学（先选课后退课的情况），则不检查文件也不给出任何错误提示信息
7-b16 -replace 100384 4-2.cpp 4-2-m.cpp	将课号为 100384 的所有同学的 4-2.cpp 文件中的 main 函数用./source/4-2-m.cpp 中的 main 函数替换 1、-replace 后面的第 1 参数是课号，第 2 参数是要替换的源程序文件名，第 3 参数是替换 main 函数所在的源程序文件名（一定放在./source 下），课号最大长度为 15 位，两个源程序文件名最大长度均为 31 2、假设 4-2.cpp 中的 main 函数一定在最后 3、假设 4-2-m.cpp 中只有一个 main 函数，并且保证一定放在 source 目录下，否则提示“替换文件不存在”即可 4、替换完成则给出提示信息，若某些同学的 4-2.cpp 不存在，则给出“4-2.cpp 未提交”即可，继

	续执行后续学生的 main 函数替换 5、-replace 不允许跟 all

注：1、-replace 和 -checktitle 不会同时出现

【作业要求：】

- 用 C++语言的文件读写方式完成
  - 程序中不允许使用 FILE *及使用与其相关的函数
  - 所有对文件操作的函数都要求与 fstream/ifstream/ofstream 流对象相关，允许使用相关但课上未讲过的流函数
  - 标准输入输出不限制使用 cin/cout 或者 printf/scanf
- 用 C 语言的文件读写方式完成
  - 程序中不允许包含 fstream 及使用与其相关的任何流对象
  - 所有对文件操作的函数都要求与 FILE *有关的，允许使用相关但课上未讲过的函数
  - 标准输入输出不限制使用 cin/cout 或者 printf/scanf

17、 一组配置文件读写函数

【问题描述：】在 Windows 和 Linux 操作系统中，很多程序有配置文件，用来设定程序运行过程中的各个选项，配置文件的结构说明如下：

```
[VideoProperties]
Title=属性设置
Title_V=10

[SpecialEffect]
Title=特效
EffectBlock=12.3
ZoomBlock=

[FaceTrack]
Title=人脸追踪
FaceTrackingBlock=y
```

★ 配置文件分为若干组，每组用[***]表示组名，
组名各不相同

★ 每组有若干项，每项的基本格式是“项目名=值”，
同组的项目名不相同，不同组可能相同
- 项目名也可能是中文
- 每个项目一行，不允许多项目一行

★ 值的可能取值有：
整数、浮点数、字符串、字母、空
- 字符串可能为中文

【函数定义(每个定义均为两个，一个是C++方式，一个是C方式)：】

★ int group_add(fstream &fp, const char *group_name)

★ int group_add(FILE *fp, const char *group_name)

示例说明：假设在main函数中调用 group_add(fp, "test");，则表示在配置文件的加入[test]组，组中暂时无内容

- 增加成功返回1，否则返回0
- 如果[test]组已存在，则不能重复增加，直接返回0即可
- 加入的组放在文件的最后即可

★ int group_del(fstream &fp, const char *group_name)

★ int group_del(FILE *fp, const char *group_name)

示例说明：假设在main函数中调用 group_del(fp, "test");，则表示在配置文件中删除[test]组及该组下存在的全部项

- 删除成功返回1，否则返回0
- 如果[test]组不存在，直接返回0即可
- 如果[test]组重复存在（例如：手工修改使两组同名），则删除位置靠前的一组即可

★ int item_add(fstream &fp, const char *group_name, const char *item_name, const void *item_value, const enum ITEM_TYPE item_type)

★ int item_add(FILE *fp, const char *group_name, const char *item_name, const void *item_value, const enum ITEM_TYPE item_type)

示例说明：1、假设在main函数中有：int i = 12345;

item_add(fp, "test", "起始值", &i, TYPE_INT);

则表示在配置文件的[test]组最后加入 "起始值=12345" 项

2、假设在main函数中有：double d = 123.45;

item_add(fp, "test", "起始值", &d, TYPE_DOUBLE);

则表示在配置文件的[test]组最后加入 "起始值=123.45" 项（如果double型）

3、假设在main函数中有：char *s="今天是个好日子";

item_add(fp, "test", "起始值", s, TYPE_STRING);

则表示在配置文件的[test]组最后加入 "起始值=今天是个好日子" 项

4、假设在main函数中有：char c = 'Y';

item_add(fp, "test", "起始值", &c, TYPE_CHARACTER);

则表示在配置文件的[test]组最后加入 "起始值=Y" 项

5、假设在main函数中有：item_add(fp, "test", "起始值", NULL, TYPE_NULL);

则表示在配置文件的[test]组最后加入 "起始值=" 项

- 增加成功返回1，否则返回0
- 如果[test]组不存在，直接返回0即可

- 如果[test]组中"起始值"已存在，则不能重复增加，直接返回 0 即可
- item 的类型定义为：enum ITEM_TYPE { TYPE_INT, TYPE_DOUBLE, TYPE_STRING, TYPE_CHARACTER, TYPE_NULL};
- 因为数据类型错误导致运行出错，不算错误（例如：TYPE_STRING 给了一个 double 地址，因为无尾零将配置文件写乱；TYPE_INT 给了一个 short 型地址，导致写入的 int 型数据不正确等）

★ int item_del(fstream &fp, const char *group_name, const char *item_name)
★ int item_del(FILE *fp, const char *group_name, const char *item_name)
示例说明：假设在 main 函数中有：item_add(fp, "test", "起始值");，则表示在配置文件的[test]组中删除"起始值=***" 项

- 删除成功返回 1，否则返回 0
- 如果[test]组不存在，直接返回 0 即可
- 如果[test]组存在，但要删除的项（例：起始值）不存在，则直接返回 0 即可
- 如果[test]组存在，但要删除的项（例：起始值）重复存在，则删除位置靠前的一项，返回 1 即可

★ int item_update(fstream &fp, const char *group_name, const char *item_name, const void *item_value, const enum ITEM_TYPE item_type)
★ int item_update(FILE *fp, const char *group_name, const char *item_name, const void *item_value, const enum ITEM_TYPE item_type)
示例说明：1、假设在 main 函数中有：int i = 12345;
item_update(fp, "test", "起始值", &i, TYPE_INT);
则表示在配置文件的[test]组将"起始值=***"项更新为"起始值=12345"；
若"起始值"项不存在，则在最后加入"起始值=12345"项

2、假设在 main 函数中有：double d = 123.45;
item_update(fp, "test", "起始值", &d, TYPE_DOUBLE);
则表示在配置文件的[test]组将"起始值=***"项更新为"起始值=123.45"；
若"起始值"项不存在，则在最后加入"起始值=123.45"项

3、假设在 main 函数中有：char *s="今天是个好日子";
item_update(fp, "test", "起始值", s, TYPE_STRING);
则表示在配置文件的[test]组将"起始值=***"项更新为"起始值=今天是个好日子"；
若"起始值"项不存在，则在最后加入"起始值=今天是个好日子"项

4、假设在 main 函数中有：char c = 'Y';
item_update(fp, "test", "起始值", &c, TYPE_CHARACTER);
则表示在配置文件的[test]组将"起始值=***"项更新为"起始值=Y"；
若"起始值"项不存在，则在最后加入"起始值=Y"项

5、假设在 main 函数中有：item_update(fp, "test", "起始值", NULL, TYPE_NULL);

则表示在配置文件的[test]组将"起始值=***"项更新为"起始值="；
若"起始值"项不存在，则在最后加入"起始值="项

- 删除更新/新增成功返回 1，否则返回 0
- 如果[test]组不存在，直接返回 0 即可
- 如果[test]组存在，但要更新的项（例：起始值）不存在，则加入在该组最后，返回 1 即可
- 如果[test]组存在，但要删除的项（例：起始值）重复存在，则只更新位置靠前的一项，返回 1 即可

★ int item_get_value(fstream &fp, const char *group_name, const char *item_name, void *item_value, const enum ITEM_TYPE item_type)

★ int item_get_value(FILE *fp, const char *group_name, const char *item_name, void *item_value, const enum ITEM_TYPE item_type)

示例说明：1、假设在 main 函数中有：int i;

```
item_get_value(fp, "test", "起始值", &i, TYPE_INT);
```

如果配置文件的[test]组有"起始值=12345"，则调用后 i 值是 12345

2、假设在 main 函数中有：double d;

```
item_get_value(fp, "test", "起始值", &d, TYPE_DOUBLE);
```

如果配置文件的[test]组有"起始值=123.45"，则调用后 d 值是 123.45

3、假设在 main 函数中有：char s[80];

```
item_get_value(fp, "test", "起始值", s, TYPE_STRING);
```

如果配置文件的[test]组有"起始值=今天是个好日子"，则调用后 s 值是"今天是个好日子"

4、假设在 main 函数中有：char c;

```
item_get_value(fp, "test", "起始值", &c, TYPE_CHARACTER);
```

如果配置文件的[test]组有"起始值=Y"，则调用后 c 的值是'Y'

5、假设在 main 函数中有：任意类型 var;

```
item_get_value(fp, "test", "起始值", &var, TYPE_NULL);
```

如果配置文件的[test]组有"起始值=***"（任意项），则调用后 var 的值不会有任何变化

- 取值成功返回 1，否则返回 0
- 如果[test]组不存在，直接返回 0 即可，不要改变传入的 void *item_value 的值
- 如果[test]组中"起始值"不存在，直接返回 0 即可，不要改变传入的 void *item_value 的值
- 当取值类型为 TYPE_NULL 时，不做任何操作，返回 1 即可，也不能改变传入的 void *item_value 的值
- 因为数据类型错误导致运行出错，不算错误（例如：TYPE_INT 给了一个 short 型地址，导致系统弹窗报错；TYPE_STRING 给个一个未指向确定空间的 char *值/不足以容纳整个字符串的一维字符数组等）

【要求:】1、用 C++/C 语言的文件读写方式分别完成

2、C++方式的作业由 7-b17-1.h/7-b17-1-sub.cpp/7-b17-1-main.cpp 组成，其中 7-b17-1.h 中存放函数声明及其它需要的内容，7-b17-1-sub.cpp 中存放这组函数的具体实现过程，7-b17-1-main.cpp 中放 main 函数，具体内容为打开/关闭配置文件、测试用例等；C 方式的作业由 7-b17-2.h/7-b17-2-sub.cpp/7-b17-2-main.cpp 组成，含义同

3、main 函数中测试用例设计的好坏，思维缜密程度占本题分数的 30%

4、整个 main 函数，仅允许开始时打开文件，测试用例运行结束后关闭文件（即文件仅允许打开/关闭一次，中间的任何操作不允许关闭后再次打开，**否则得分为 0**）

5、所有的字符串形式（TYPE_STRING），均不含空格、回车、TAB 键等控制字符

【作业要求:】

1、**4 月 12 日前**网上提交本次作业

2、每题所占平时成绩的具体分值见网页

3、超过截止时间提交作业则不得分