学院	辅修专业	学号	姓名
化学学院	计算机科学与技术	16190311	李骁睿

编译环境：

gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)

Linux 5.11.0-27-generic x86_64 GNU/Linux

系统设计：

本系统是一个 Unix Shell 风格的学生选课信息管理程序，使用链表作为记录的基本数据结构。主要分 3 大模块：指令解析和用户交互模块、账户模块、数据模块。其中，主函数 main()和主循环 interactiveShell()负责指令解析和用户交互；login.c 包含了所有用户创建、用户登录、用户身份识别、登录验证、信息保存的功能；lnklist.c 和 curriculum.c 包含了底层数据类型双向链表的具体实现以及数据的显示、增添、删除等操作。

用户交互模块可以通过 int main(int argc, char** argv)从命令行获取参数从而执行指令或者进入交互模式。（为了方便命令行执行，本系统从一开始便为命令行模式设计，由于精力关系只能先实现交互模式。）当执行 eduadmin -i 时便进入交互 shell 模式。-h 或者--help 参数可以打印帮助信息。以下为程序中的帮助信息。

```
Usage: eduadmin [Options] [Command args ...]
Options:
    -h, --help
        Print this help message.
Interactive shell commands:
    help, ?
        Show this help message.

    quit, exit
        Save data and quit.

    add [user|admin|course|room]
        Add new records.

    list [user|course|teacher|student|room]
        Show all records of users, courses, teachers, students and rooms.

    show [student|course|teacher]
        Show more detailed information for specified UID.

    enroll
        Enrool to a course. Only students are able to use this command.

    disenroll
        Disnrool to a course. Only students are able to use this command.

    register
        Register to a course. Only teachers can use this command.

    sort [user|course|room] by [name|id|credit|capacity]
        Sort records by specified field name.

    me
        Show the detailed information for user currently logged in. Only students
        can use this command.

    save
        Manually save all records and user information.

    logout
        Logout from user currently logged in to user "anonymous".
```

interactiveShell()函数是主循环。从 stdin 读取用户输入，通过 parser.c 中的 parseArgs(char *argv[MAX_ARGC], char *str)函数将命令切割为参数并判断命令和子命令，将参数传入对应的模块函数完成调用。

账户模块的主要函数是 getUserInfo()函数、saveLoginInfo()函数、login()函数和 createAccount(bool admin)函数。

getUserInfo 函数从目录中现有的 user 文件中以二进制形式读取所有用户的用户名、UID 和哈希值，以链表形式读入内存。saveLoginInfo()则相反。

login()函数是用户登录的关键函数。用户在提示下输入用户名和密码。之后通过用户名和密

码拼接和 MD5 Hash 生成一个 16 位的 Hash 校验值，与文件中读取的用户对应的 Hash 值相比对。若一致则通过，并将全局变量 USERID 设为文件中读取的 USERID，完成登录。为了保护登录密码，在登录时会将输入的密码字符以"*"掩盖。

createAccount()函数是用户创建函数。这个函数根据用户输入的用户名和密码，通过拼接生成 MD5，并将新账户加入到 USERLIST 链表中。USERID 与学号不同。注意到学号最大不会超过 2^28，可存储在 32bit 的无符号长整型 uint32_t 中。开头 4 位空余可以用来作为区分教师、学生。也可以设置管理权限位。账户系统拥有两个默认账户 anonymous 和 root。UID 分别为 0 和 1，对应系统的最低权限和最高权限。只有 root 可以创建具有管理权限的账户。anonymous 只允许查看课程和教室信息。saveLoginInfo()可以在退出时或使用 save 命令手动调用。为了避免直接通过读取密码文件的方式获得他人的登录信息，密码以 MD5 的形式与用户名和 UID 一同以二进制形式写入 passwd 文件中。当登录信息文件不存在时，程序会提供一个初始化帮助函数 initUserProfile()帮助创建 root 账户和 passwd 文件。

用户数据模块在 curriculum.c 中，底层的数据结构为双向链表，在 lnklist.c 中实现。lnklist.c 中实现了双向链表的初始化、释放和节点的增加、删除、交换、遍历、查询等操作。后面采用的快速排序算法对链表中的两项直接进行交换操作较为繁琐且容易出错，因此将链表设计为指针的形式指向数据储存区，交换时只需要交换指针，读取数据时需要指针类型的强制转换。

seekNode()函数设计为传入一个链表信息的结构体、一个返回 bool 值的函数指针用于匹配查询值、和一个指向被比较数据的 void*指针。qsortcore()是快速排序算法的核心实现。节点的比较也通过传入函数指针的方式进行，因此可以以平均 O(nlgn)的时间复杂度进行各种排序，也可以传入一个布尔型的 reverse 参数来指定是否倒序排序。curriculum.c 中包括上述函数所需的各种比较函数与匹配函数。initData()函数将储存链表用户信息、课程信息、教室信息等的全局变量结构体初始化。loadData()则从 courses，enrollments，rooms，registries 四个文件中读取信息并添加在链表中，并处理相互之间指针的链接关系。一系列 userxxxx 函数负责获取并验证用户输入的合法性，传入 addxxxx 函数中进行真正的添加操作。enroll 和 disenroll 函数负责学生选课和退选。每一个函数中都会有对当前用户操作权限的检验。且不同权限的账户对用户信息输出的级别不同。例如只有具有管理权限的账户才能查看具体的选课学生信息。普通账户只能显示选课人数等。

me()函数是为学生提供的查看并计算当前账户选课情况统计信息的函数。显示学生姓名、学号、选课列表、总学分、必修学分、选修学分等。一系列 listxxxx 函数用于遍历链表输出信息，便于用户检索。一系列 showxxxx 函数用于显示给定 USERID/COURSEID/ROOMID 的数据详细信息。课程 ID 的设计也是与用户 ID 类似，最高位被设计为区分选修与必修的位。这导致必修课的课程 ID 必然大于 2147483647。

关键代码：

常量：

```
#ifndef LIMITS
#define LIMITS

#define MAX_LEN 256
#define MAX_ARGC 16
#define MD5_LEN 16
#define MAX_TEACHERS 4
#define MAX_TRAILS 4

#endif
```

全局变量(globals.h):

```c
#ifndef GLOBALS
#define GLOBALS
#include "login.h"
#include "lnklist.h"
#include "limits.h"
#include <stdint.h>
#include <stdio.h>

char USERNAME[MAX_LEN];
uint32_t USERID;
extern uint8_t _binary_HELP_end;
extern uint8_t _binary_HELP_start;

DLnklist *USERLIST;

DLnklist *TEACHERS;
DLnklist *STUDENTS;
DLnklist *ENROLLMENTS;
DLnklist *COURSES;
DLnklist *ROOMS;
DLnklist *REGISTRYS;

#endif      You, 上周 • 写不动了...
```

一些结构体定义(curriculum.h):

```c
You, 6天前 | 1 author (You)
typedef struct CTime
{
    uint8_t weekstart;
    uint8_t weekend;
    uint8_t weekday;
    uint8_t timestart;
    uint8_t timeend;
} CTime;

You, 6天前 | 1 author (You)
typedef struct Room
{
    uint32_t roomid;
    char name[MAX_LEN];
    uint32_t capacity;
    DLnklist courses;
} Room;

You, 6天前 | 1 author (You)
typedef struct Enrollment
{
    uint32_t studentid;
    uint32_t courseid;
} Enrollment;

You, 6天前 | 1 author (You)
typedef struct Registry
{
    uint32_t courseid;
    uint32_t teacherid;
} Registry;
```

```
You, 3天前 | 1 author (You)
typedef struct Course
{
    uint32_t courseid;
    char name[MAX_LEN];
    char description[MAX_LEN];
    float credits;
    DLnklist teachers; // teachers
    uint32_t capacity;
    DLnklist enrollment; // students
    CTime time;
    Room *room;
} Course;

You, 6天前 | 1 author (You)
typedef struct Teacher
{
    uint32_t teacherid;
    char name[MAX_LEN];
    DLnklist registry; // courses
} Teacher;

You, 6天前 | 1 author (You)
typedef struct Student
{
    uint32_t studentid;
    char name[MAX_LEN];
    DLnklist enrollment; // courses
} Student;
```

参数分析(parser.c)

```
int parseArgs(char *argv[MAX_ARGC], char *str)
{
    int argc = 0;
    int len = strlen(str);
    for (int i = 0; i < len && argc < MAX_ARGC;)
    {
        // First check if alnumoper
        if (!isChars(str[i], WHITESPACE))
        {
            argv[argc] = &str[i]; // Yes, a new argument, mark start position and skip all alnumoper
            for (; i < len && !isChars(str[i], WHITESPACE); i++)
                ;
            argc++; // Commit new argument
        }
        else
        { // Not alnumoper, no arguments, break
            break;
        }      You, 上周 • parser …
        // Skipped all alnumoper, now check if space/tab
        if (isChars(str[i], SPACE))
        {
            str[i] = '\0'; // Replace space/tab with null terminator
            for (; i < len && isChars(str[++i], SPACE);)
                ; // Skip all spaces/tabs
        }
        else // not space/tab, break
        {
            break;
        }
    }
    return argc;
}
```

生成 MD5 Hash(login.c)

```c
int loginHash(char *username, char *password, char hash[MD5_LEN]) // generates md5 from username and
{
    int un_len, pw_len = 0;
    un_len = strlen(username);
    pw_len = strlen(password);
    uint8_t buf[MD5_LEN] = {0};
    strcpy(buf, username);
    strcpy(buf + un_len, ":");
    strcpy(buf + un_len + 1, password); // Concatenate username and password
    md5(buf, strlen(buf), hash); // calculate md5sum
}
```

(md5 代码来源 GitHub　scottjg/dracd/md5.c)

链表实现(lnklist.h 和 lnklist.c):

```c
typedef struct DNode // linked list node
{
    void *data;
    struct DNode *next;
    struct DNode *prev;
} DNode;

...
typedef struct DLnklist // save information of a linked list
{
    DNode *head;
    DNode *tail;
    DNode *ptr;
    uint32_t count;
} DLnklist;

DLnklist *makeDLinkedList();
int initDLinkedList(DLnklist *dlist);

int appendNode(DLnklist *ll, void *data);
int insNode(DLnklist *ll, void *data); // insert after ptr
int insHeadNode(DLnklist *ll, void *data);
int moveNext(DLnklist *ll);
int delNode(DLnklist *ll);
int seekNode(DLnklist *ll, bool (*criterion)(void *, void *), void *data);
int nodeqsort(DLnklist *ll, bool (*criterion)(void *, void *), bool reverse);
int qsortcore(DNode *head, DNode *tail, bool (*criterion)(void *, void *), bool reverse);
int clearList(DLnklist *ll);
#endif
```

节点增加

```c
int appendNode(DLnklist *ll, void *data)
{
    if (ll->count != 0)
    {
        DNode *tmp = ll->ptr;
        ll->ptr = ll->tail;
        insNode(ll, data);
        ll->ptr = tmp;
        return ll->count;
    }
    else
    {
        ll->ptr = ll->tail;
        insNode(ll, data);
        return ll->count;
    }
}
```

节点插入

```c
int insNode(DLnklist *ll, void *data) // insert after ptr
{
    DNode *node = calloc(1, sizeof(DNode));
    node->data = data; // for other functions to fill in
    if (ll->count == 0)
    {
        node->next = NULL;
        node->prev = NULL;
        ll->head = node;
        ll->tail = node;
        ll->ptr = node;
        ll->count++;
        return ll->count;
    }
    node->next = ll->ptr->next;
    node->prev = ll->ptr;
    if (ll->ptr != ll->tail)
    {
        ll->ptr->next->prev = node;
    }
    else // nullptr -> tail
    {
        ll->tail = node;
        // printf("%p\n", ll->ptr->data);
    }
    ll->ptr->next = node;
    ll->ptr = node;
    ll->count++;
    return ll->count;
}
```

节点搜索

```c
int seekNode(DLnklist *ll, bool (*criterion)(void *, void *), void *data)
{
    if (ll->ptr == NULL)
    {
        return 1; // error of empty list
    }
    ll->ptr = ll->head;
    for (; !criterion(ll->ptr->data, data); moveNext(ll))
    {
        if (ll->ptr == ll->tail) // end of list
        {
            return -1; // not found
        }
    }
    return 0; // found
}
```

节点交换和 quick sort

```c
int nodeqsort(DLnklist *ll, bool (*criterion)(void *, void *), bool reverse)
int swapNode(DNode *a, DNode *b)
{
    if (a != b)
    {

        void *tmp = a->data;
        a->data = b->data;
        b->data = tmp;
    }
    return 0;
}
int qsortcore(DNode *head, DNode *tail, bool (*criterion)(void *, void *), bool reverse)
{
    if (head == tail)
    {
        return 0;
    }
    DNode *left = head;
    DNode *right = tail;
    while (left != right)
    {
        if (reverse ^ criterion(left->data, left->next->data))
        {
            swapNode(left, left->next);      You, 3天前 • 20220301 …
            left = left->next;
        }
        else
        {
            swapNode(left->next, right);
            right = right->prev;
        }
    }
    qsortcore(head, left, criterion, reverse);
    if (left != tail)
    {
        qsortcore(left->next, tail, criterion, reverse);
    }
    return 0;
}
```

UID 生成(login.c createAccount()函数节选)

```c
    while (getchar() != '\n') // Clear stdin
        ;
    printf("Creating user...\n");
    uid = (sub_id & 0x0FFFFFFF) | (!is_student << 31) | (admin << 30);
    if (is_student)
    {
        addStudent(username, sub_id);
    }
    else
    {
        addTeacher(username, sub_id);
    }
    createLogin(username, password, uid);
    printf("User created!\n");
    return 0;
```

UID 和学号转换以及身份识别

```
bool isTeacher(uint32_t userid)
{
    return ((userid & 0x80000000) >> 31) && (!isAnonymous(userid));
}

bool isStudent(uint32_t userid)
{
    return (~(userid & 0x80000000) >> 31) && (!isAnonymous(userid));
}

bool isRoot(uint32_t userid)
{
    return userid == 1;
}

bool isAdmin(uint32_t userid)
{
    return ((userid & 0x40000000) >> 30) || isRoot(userid);
}

bool isAnonymous(uint32_t userid)
{
    return userid == 0;
}

uint32_t uid2subid(uint32_t uid)
{
    return uid & 0x0FFFFFFF;
}
```

测试结果：

用户登录、登出

```
❯ ./eduadmin
Welcome to use EDUAdmin.
No arguments, interactive mode.
[anonymous]> login root
Password: *************
Login success!
[root]> login root
You are already logged in.
[root]> logout
Logged out.
[anonymous]>
```

学生登陆、信息展示、课程浏览与选课

```
❯ ./eduadmin
Welcome to use EDUAdmin.
No arguments, interactive mode.
[anonymous]> login Merlyn
Password: ************
Login success!
[Merlyn]> list course
Courses:
0: Python
        none

1: Computer_Vision
        none

2147483650: C_Language
        none

2147483651: Operating_System
        none

2147483652: Computer_Network
        none

5: Data_Mining
        none

6: Java
        none

7: Object_Oriented_Programming
        none

8: Network_Security
        none

2147483657: Computer_Systems
        none

[Merlyn]> enroll
Courseid: 6
Enrolled course id 6 successfully.
[Merlyn]> enroll
Courseid: 5
Enrolled course id 5 successfully.
[Merlyn]> enroll
Courseid: 6
Error: already enrolled.
[Merlyn]>
```

```
[Merlyn]> me
Information about Merlyn:
Studentid: 16190311

Enrollments:
6: Java 3.000000
5: Java 3.000000

Total: 6.000000
Compulsory: 0.000000
Elective: 6.000000
[Merlyn]>
```

```
[anonymous]> me
You are not a student.
[anonymous]>
```

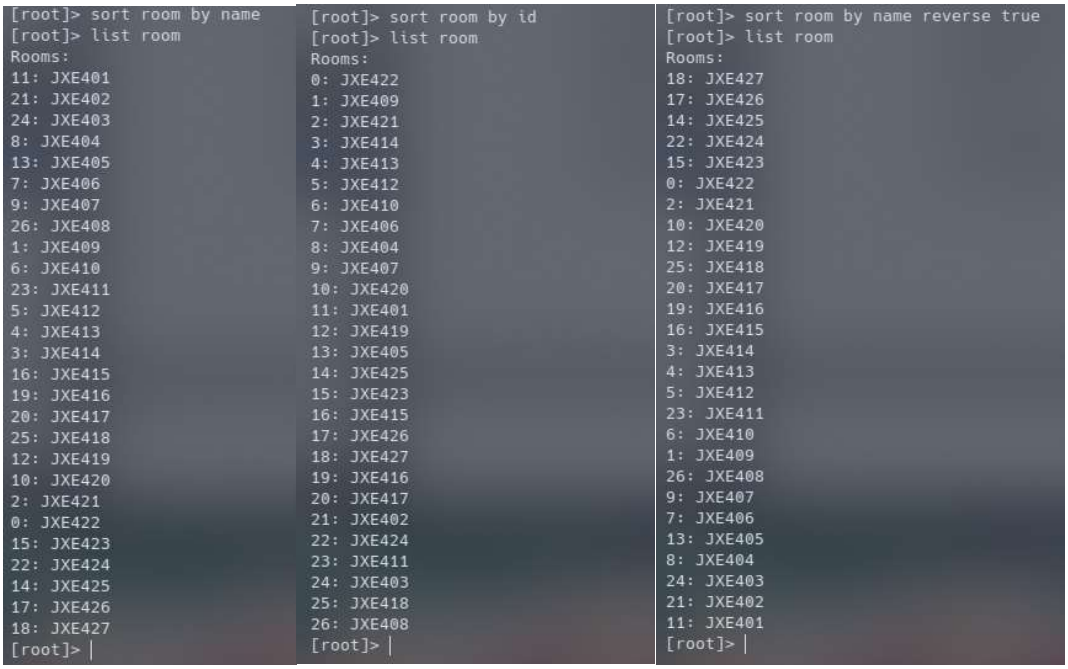
Root 账户增加课程、用户、管理员

```
[root]> add course
Course name: Topology
Description: none
Credits: 3
Roomid: 0
Capacity: 30
Type: [Compulsory(C), Elective(E)] E
Week start: 1
Week end: 14
Weekday (1-7): 3
Class start at: 5
Class end at: 8
Error: time conflict. Retry.
Week start: 1
Week end: 14
Weekday (1-7): 5
Class start at: 5
Class end at: 8
Course added.
[root]>
```

```
[root]> add user
Username: Jacob
Password: Jacob
Password Again: Jacob
Account type Teacher/Student [T/S]: S
Student ID: 16190312
Creating user...
User created!
[root]>
```

```
[root]> add admin
Username: Steve
Password: passwd
Password Again: passwd
Account type Teacher/Student [T/S]: T
Teacher ID: 11112222
Creating user...
User created!
[root]>
```

排序



系统特色：

本系统在用户权限和账户安全方面做了额外考量。采用哈希值的形式存储密码完全避免了密码泄露。并且以用户名作为 salt 避免使用常见密码的哈希进行暴力破解。在管理员正确设置密钥文件权限的前提下可以达到无法破解密码从而避免篡改信息的目的。并且为系统设计了不同的权限和 UID 使得不同用途的账户可以在同一个系统登录。使用链表可以快速动态的增加删除记录。同时使用快速排序算法使排序过程更加高效。教师和学生以链表的方式存储的使得列表长度可以动态增加。