* 1. **课程设计任务、要求、目的**
  2. **课程设计任务**

访问控制技术：建立基于访问控制表的系统安全管理。

* 1. **课程设计目的和要求**

系统应该包含两个部分，一个部分是按内核代码原则设计的内核系统，由一系列的函数组成；另一个部分是演示系统，调用内核系统的相应函数，以让其运行，同时提供系统的展示界面，可以是GUI或者字符界面，以展示系统的运行状态，显示系统的关键数据结构的内容。

具体包括：

1) 建立以用户组为管理域的访问控制表基本数据结构；

2) 模拟操作系统中的若干对象，并可添加到访问控制表中；

3) 建立操作系统的用户模型；

4) 模拟用户登录及切换，访问对象时的受限情况。

* 1. **开发环境**

操作系统：Ubuntu 20.04

编译器：GNU C++ Compiler 9.3.0

* 1. **相关原理及算法**

**3.1 访问控制**

访问控制（Access Control）指系统对用户身份及其所属的预先定义的策略组限制其使用数据资源能力的手段。通常用于系统管理员控制用户对服务器、目录、文件等网络资源的访问。访问控制是系统保密性、完整性、可用性和合法使用性的重要基础，是网络安全防范和资源保护的关键策略之一，也是主体依据某些控制策略或权限对客体本身或其资源进行的不同授权访问。

访问控制的主要目的是限制访问主体对客体的访问，从而保障数据资源在合法范围内得以有效使用和管理。为了达到上述目的，访问控制需要完成两个任务：识别和确认访问系统的用户、决定该用户可以对某一系统资源进行何种类型的访问。

访问控制包括三个要素：主体、客体和控制策略。

（1）主体S（Subject）。是指提出访问资源具体请求。是某一操作动作的发起者，但不一定是动作的执行者，可能是某一用户，也可以是用户启动的进程、服务和设备等。

（2）客体O（Object）。是指被访问资源的实体。所有可以被操作的信息、资源、对象都可以是客体。客体可以是信息、文件、记录等集合体，也可以是网络上硬件设施、无限通信中的终端，甚至可以包含另外一个客体。

（3）控制策略A（Attribution）。是主体对客体的相关访问规则集合，即属性集合。访问策略体现了一种授权行为，也是客体对主体某些操作行为的默认。

**3.2 用户账户控制**

用户帐户控制（User Account Control，简写作UAC）是微软公司在其Windows Vista及更高版本操作系统中采用的一种控制机制。其原理是通知用户是否对应用程序使用硬盘驱动器和系统文件授权，以达到帮助阻止恶意程序（有时也称为“恶意软件”）损坏系统的效果。

UAC的四个级别：

1. 出现以下情况时始终通知我：

• 程序试图安装软件或更改我的计算机

• 我更改了Windows设置

2. 仅在程序尝试对我的计算机进行更改时通知我

• 当我对Windows设置进行更改时不要通知我。

3. 仅当程序尝试更改计算机时通知我（不降低桌面亮度）

• 当我对Windows设置进行更改时不要通知我。

4. 出现以下情况时始终不要通知我（禁用UAC）：

• 程序试图安装软件或更改我的计算机

• 我更改了Windows设置

* 1. **系统结构和主要的算法设计思路**

**4.1 系统结构**

系统分为内核代码和测试代码两部分。

系统内核部分有两个文件access\_lib.h和access\_lib.cpp，编译后生成access\_lib.o目标代码文件，而后将其打包生成libaccess.a静态库，由测试部分代码demo.h和demo.cpp调用，编译得到demo.o目标代码文件，并与静态库一起链接生成demo可执行文件，以上过程可由makefile工程文件完成。

demo可执行文件可以运行测试样例，测试样例中包含一系列的指令和命令。程序在运行过程中会执行测试样例中的命令，生成相应的文件，在控制台打印script.sh的运行结果和audit.txt文件内容。

系统模拟了用户的登录和登出操作、用户和用户组管理、用户权限的更改、用户访问受限等操作，模拟了操作系统中的文件和程序两个对象，并实现了它们的新建、读取、写入和运行操作，在程序对象上借鉴引入了简化的UAC机制，可执行bash shell命令（由操作系统完成），最后以文件形式输出结果。

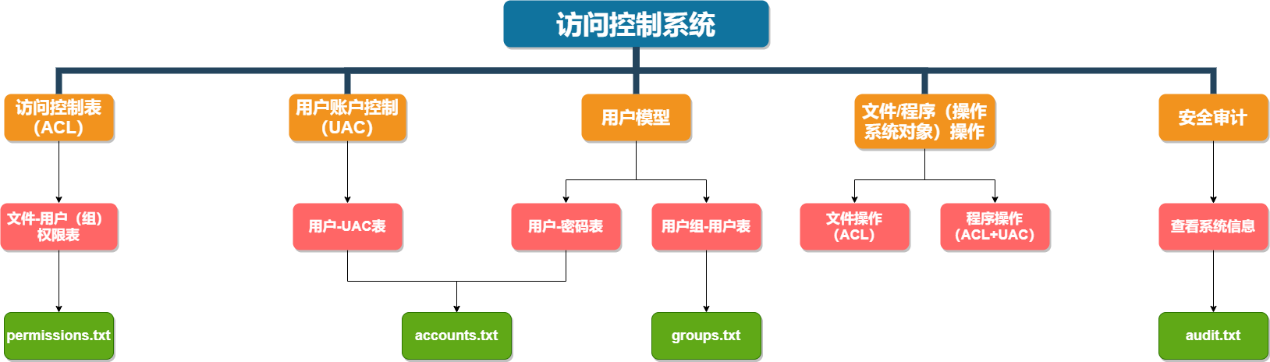


图 1 系统架构图

**4.2 主要的算法设计思路**

系统使用命令行来实现访问控制功能，每条命令的语法最多有4个组成部分。系统中实现的全部命令如下：

1. net user [username] [password] 创建一个用户并设置密码

username 用户名

password 密码

1. net group [group-name] 创建一个用户组

group-name 用户组名

1. net group [group-name] [username] 将用户添加到组中

group-name 用户组名

username 用户名

1. del user [username] 删除一个用户

username 用户名

1. del group [group-name] 删除一个用户组

group-name 组名

1. login [username] [password] 登录用户

username 用户名

password 密码

1. logout 登出当前用户
2. xcacls [filename] [arg] [new-permissions] 更改权限

filename 用户名

arg

/E 新增权限

/P 替换权限

/D 拒绝权限（无权限）

new-permissions 新权限

1. create [filename] 新建一个文件

filename 文件名

1. read [filename] 读取一个文件

filename 文件名

1. write [filename] [content] 向一个文件写入文本

filename 文件名

content 要写入的文本内容

1. execute [filename] 执行（打开）文件

filename 文件名

1. program create [filename] 新建一个程序

filename 文件名

1. program read [filename] 读取一个程序

filename 文件名

1. program write [filename] [content] 向一个程序中写入文本

filename 文件名

content 要写入的文本内容

1. program execute [filename] 执行一个程序

filename 文件名

1. uac [arg] 改变当前用户的UAC状态

arg

Always 始终通知

Change 只在改变设置时通知

Never 永不通知

1. yes 确认
2. end 结束

系统中规定必须有Administrators和Users两个用户组作为管理员组和用户组，和一个用户名为admin的管理员，该管理员在Administrators组中，Users组包括除管理员之外的所有用户。因此，测试样例的前四行必须是：

net user admin [password]

net group Administrators

net group Administrators admin

net group Users

系统会在初始阶段检测前四行，如果前四行不符合上述输入，则会报告错误，拒不执行下面的命令。

所有的指令、访问控制表、账户密码、UAC表等空间占用较小的信息在系统中以内存形式存储；日志信息则直接写入audit.txt文件中，不存储在内存中。

然后系统会根据上述的命令格式匹配参数，调用相应的函数，各函数中将操作记录和输出的结果写入audit.txt日志文件中。最后以end命令结束。end命令会生成accounts.txt、groups.txt、files.txt三个文件，分别代表用户-密码-UAC表、用户组-用户表、文件-用户权限表。

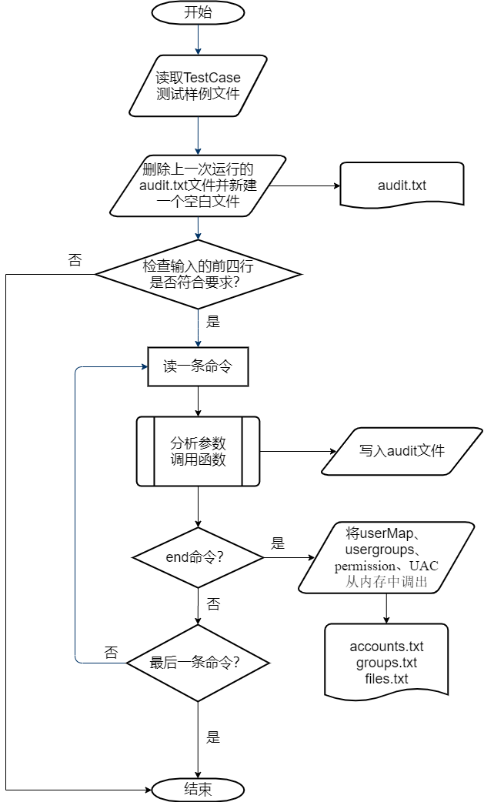


图 2 主程序流程图

* 1. **程序实现---主要数据结构**

1. 指令数组：vector<string> instructions

所有指令（instructions）存储在一个vector向量中，在程序开始运行时直接从TestCase文本中读取，vector中的每一个string字符串表示一行指令。

1. 文件-用户（组）权限字典：map<string, vector<string>> permissions

用户（组）权限以map数据结构存储。关键字（key）部分是一个string字符串，表示文件名（filename）；值（value）部分是一个vector向量，第一个string字符串表示这个文件的拥有者（owner）的用户名（username），其余的每一个string字符串表示用户（组）权限，它由用户（组）名（username/group name）和权限值（permission value）组成，并以冒号（:）分隔开。

权限值列举如下：

F 所有权限（All permissions）

R 读权限（Read permission）

W 写权限（Write permission）

X 执行权限（Execute permission）

D 无权限（Deny permission）

例如，file1.txt alice (alice:F,Administrators:F)表示该文件的拥有者是alice，用户alice和用户组Administractors拥有对该文件的所有权限。

用户在创建文件时，默认设置为拥有者和管理员组（Administractors）拥有该文件的所有权限，另外如果是管理员创建了该文件，则用户组Users对文件拥有读权限。使用xcacls命令可更改权限（只有文件拥有者和管理员有权限更改）。

1. 用户组-用户字典：map<string, vector<string>> usergroups

用户组-用户关系以map数据结构存储。关键字部分是一个string字符串，表示用户组名（group-name）；值部分是一个vector向量，vector中的每一个string字符串表示一个用户名（username）。

系统初始状态有两个用户组Administractors和Users，所有管理员在Administrators组中，除管理员之外的所有用户在Users组中。

1. 用户-UAC状态字典：map<string, string> UAC

用户-UAC状态关系以map数据结构存储。关键字部分是一个string字符串，表示用户名（filename）；值部分是一个string字符串，表示UAC状态（User Account Control state）。

UAC状态及其表示的含义列举如下：

Always Always notify 始终通知

Change Notify only on change 只在改变设置时通知

Never Never notify 永不通知

在实际处理中，系统将Always和Change进行相同的处理，实质上只有两级UAC状态。

创建用户时，默认为用户设置为Change。使用uac命令可更改当前用户的UAC状态。

1. 用户-密码字典：map<string, string> userMap

用户-密码关系以map数据结构存储。关键字部分是一个string字符串，表示用户名（filename）；值部分是一个string字符串，表示密码（password）。

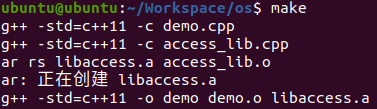
设置用户名和密码时，不允许使用一些特殊字符。

* 1. **程序实现---主要程序清单**

|  |  |
| --- | --- |
| bool processTestCase(string testname); | 主程序，处理测试样例文件中的命令，返回是否成功 |
| int checkSetup(vector[string] instructions); | 检查测试样例文件中的前四行 |
| int checkCommand(string username); | 检查用户名是否包含合法的字符 |
| int checkPassword(string password); | 检查密码是否包含合法的字符 |
| void login(string username, string password); | 用户登录 |
| void logout(); | 登出当前用户 |
| bool groupExists(string groupname); | 检查用户组是否存在 |
| bool userExists(string username); | 检查用户是否存在 |
| bool createUser(string username, string password); | 创建一个新用户 |
| bool createGroup(string groupname); | 创建一个用户组 |
| bool deleteUser(string username); | 删除一个用户 |
| bool deleteGroup(string groupname); | 删除一个用户组 |
| vector[string] getUserGroup(string username); | 得到用户所在的所有用户组 |
| bool isAdmin(string username); | 检查用户是否是管理员 |
| bool isOwner(string filename, string username); | 检查用户是否是文件的拥有者 |
| bool addToGroup(string username, string groupname); | 添加用户到用户组中 |
| bool fileExist(string filename); | 检查这个文件是否存在 |
| void createFile(string filename); | 新建一个文件 |
| bool strncasecmp(string s1, string s2); | 比较两个字符串（不区分大小写） |
| bool canContinue(string answer); | 等待用户回答是否可以继续 |
| bool contains(string s1, string s2); | 检查字符串s1是否包含字符串s2 |
| void setUAC(string username, string permissions); | 设置UAC权限 |
| string getUAC(string username); | 返回UAC权限 |
| string getUACString(string username); | 返回UAC中Never, Change, Always所表示的含义 |
| void setPermissions(string filename); | 根据文件名设置默认权限 |
| void writeFile(string filename, string text); | 写入文件 |
| void readFile(string filename); | 读取文件 |
| void executeFile(string filename); | 执行文件 |
| bool canWrite(string userPermissions); | 检查一个用户是否有权限写入 |
| bool canRead(string userPermissions); | 检查该用户是否有权限读取 |
| bool canExecute(string userPermissions); | 检查该用户是否有权限执行 |
| string getPermissions(string filename, string username); | 根据文件名和用户名得到对应的最高权限 |
| void editPermissions(string filename, string newPermissions); | 添加新的权限 |
| void replacePermissions(string filename, string newPermissions); | 替换已有的权限 |
| void denyPermissions(string filename, string newPermissions); | 设置拒绝权限 |
| void log(string text); | 将日志记录到audit.txt文件中 |
| void programExecute(string filename); | 执行程序 |
| bool isRestrictedName(string filename); | 检查文件名是否是限制中的文件名 |

* 1. **程序运行的主要界面和结果截图**

编译工程



1. 测试用例TestCase-1：文件操作和访问受限情况

TestCase-1 文件内容

|  |  |
| --- | --- |
| net user admin huy\*a98  net group Administrators  net group Administrators admin  net group Users  login admin huy\*a98  net user alice live4u\_no  net user bob gonelove2m  net group Students  net group Students alice  net group Students bob  net user tom geek\_247  logout  login alice live4u\_no  create file1.txt  write file1.txt Text from Alice in file1  create file2.txt  write file2.txt Text from Alice in file2  xcacls file2.txt /E Users:R | xcacls file2.txt /E Users:R  xcacls file2.txt /E Students:RW  logout  login bob gonelove2m  write file1.txt Text from Bob in file1  read file1.txt  write file2.txt Text from Bob in file2  read file2.txt  logout  login tom geek\_247  read file1.txt  write file1.txt Text from Tom in file1  read file2.txt  write file2.txt Text from Tom in file2  xcacls file1.txt /D alice  logout  end |

audit.txt 文件内容

|  |  |  |
| --- | --- | --- |
| User admin created  Notify only on change  Group Administrators created  User admin added to group Administrators  Group Users created  User admin logged in  User alice created  Notify only on change  User bob created  Notify only on change  Group Students created  User alice added to group Students  User bob added to group Students  User tom created  Notify only on change  User admin logged out  User alice logged in | File file1.txt with owner alice and default permissions created  User alice wrote to file1.txt: Text from Alice in file1  File file2.txt with owner alice and default permissions created  User alice wrote to file2.txt: Text from Alice in file2  The ACL for file2.txt appended by alice to include Users:R  The ACL for file2.txt appended by alice to include Students:RW  User alice logged out  User bob logged in  User bob denied write access to file1.txt  User bob denied read access to file1.txt | User bob wrote to file2.txt: Text from Bob in file2  User bob read file2.txt as:  Text from Alice in file2  Text from Bob in file2  User bob logged out  User tom logged in  User tom denied read access to file1.txt  User tom denied write access to file1.txt  User tom read file2.txt as:  Text from Alice in file2  Text from Bob in file2  User tom denied write access to file2.txt  Error with xcacls: Only file owner or member of Administrators group may run command  User tom logged out |

accounts.txt 文件内容

admin huy\*a98 Notify only on change

alice live4u\_no Notify only on change

bob gonelove2m Notify only on change

tom geek\_247 Notify only on change

groups.txt 文件内容

Administrators: admin

Students: alice bob

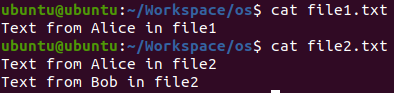
Users: alice bob tom

files.txt 文件内容

file1.txt alice (alice:F,Administrators:F)

file2.txt alice (alice:F,Administrators:F,Users:R,Students:RW)

file1.txt和file2.txt 文件内容



这个测试用例模拟了文件操作和访问受限情况，具体工作如下：

1. 管理员首先创建了alice、bob和tom三个用户，并创建了一个用户组Students，将alice和bob添加进Students组里面，管理员退出。
2. alice登录，新建了file1.txt和file2.txt两个文件，并分别写入文本，然后更改file2.txt的权限为Users用户组可读，Students用户组可写和可读。
3. bob登录，分别尝试写入和读取file1.txt和file2.txt。
4. tom登录，分别尝试写入和读取file1.txt和file2.txt，尝试更改file1.txt的权限。

以上过程的访问受限情况均在以黄色文本突出显示。查看files.txt的文件内容我们知道alice和管理员组拥有file1.txt和file2.txt的所有权限，Users用户组有Users用户组的读取权限，Students用户组有file2.txt文件的写入和读取权限。因此bob和tom在读取和写入file1.txt文件时都受到了限制，而在写入file2.txt时bob不受限制，tom受到了限制。因为tom不是文件的拥有者或管理员，所以当tom尝试更改file1.txt的权限时，系统不允许这个操作并提示错误。

1. 测试用例TestCase-2：用户管理

TestCase-2 文件内容

|  |  |
| --- | --- |
| net user admin kkan\*h2  net group Administrators  net group Administrators admin  net group Users  login admin kkan\*h2  net user john asss  net user snow sssa  net group Stuff  net group Stuff john  net group Stuff snow | logout  login joh asss  login john asss  logout  login admin kkan\*h2  del user john  logout  login john asss  logout  end |

audit.txt 文件内容

|  |  |
| --- | --- |
| User admin created  Notify only on change  Group Administrators created  User admin added to group Administrators  Group Users created  User admin logged in  User john created  Notify only on change  User snow created  Notify only on change  Group Stuff created | User john added to group Stuff  User snow added to group Stuff  User admin logged out  Login failed: invalid username or password  User john logged in  User john logged out  User admin logged in  User john deleted  User admin logged out  Login failed: invalid username or password  logout: No user logged in |

accounts.txt 文件内容

admin kkan\*h2 Notify only on change

snow sssa Notify only on change

groups.txt 文件内容

Administrators: admin

Stuff: snow

Users: snow

这个测试用例模拟了用户登录和登出，具体工作如下：

1. 管理员创建了john和snow两个用户，并创建了一个用户组Stuff，将john和snow添加进Stuff组里面，管理员退出。
2. john尝试以joh用户名登录，登陆失败，再尝试以正确的用户名和密码登录后成功登录，退出登录。
3. 管理员登录，删除了john账户。
4. john再次尝试登录时，已经无法登录。

我们可以看到john用户被删除在accounts.txt和groups.txt中都消失了。

1. 测试用例TestCase-3：程序操作和UAC机制

TestCase-3 文件内容

|  |  |  |
| --- | --- | --- |
| net user admin h4cks\_\*&/  net group Administrators  net group Administrators admin  net group Users  login admin h4cks\_\*&/  uac Always  net user john &DS4Hh  logout  login john &DS4Hh  uac Never  program create accounts.txt  program create accountNames.txt  program write accountNames.txt John  read accountNames.txt  logout  login admin H4cks\_\*&/  net group Testers  login admin h4cks\_\*&/  net group Testers  net group Testers john  program create log.txt  yes  program write log.txt Today was a good day. | yes  net user rob 237hd\_.L?  net group Testers rob  net group Programmers  net user veronica 47\*3nd\_)/d\  net group Programmers veronica  logout  login veronica 47\*3nd\_)/d\  uac Always  create script.sh  write script.sh #!/bin/bash  program write script.sh echo count to 4:  yes  xcacls script.sh /E Testers:X  logout  login rob 237hd\_.L?  uac Never  program read script.sh  program execute script.sh  logout  login veronica 47\*3nd\_)/d\  write script.sh COUNTER=0  write script.sh while [ $COUNTER -lt 5 ]; do | write script.sh echo $COUNTER  write script.sh let COUNTER=COUNTER+1  write script.sh done  read script.sh  program write script.sh echo ...Done!  yes  logout  login john &DS4Hh  program execute script.sh  logout  login admin h4cks\_\*&/  write accountNames.txt Rob  write accountNames.txt Veronica  write log.txt Finished adding account names.  read log.txt  program read accountNames.txt  yes  logout  end |

audit.txt 文件内容

|  |  |  |
| --- | --- | --- |
| User admin created  Notify only on change  Group Administrators created  User admin added to group Administrators  Group Users created  User admin logged in  Always notify  User john created  Notify only on change  User admin logged out  User john logged in  Never notify  Error: Cannot use accounts.txt as filename  File accountNames.txt with owner john and default permissions created  User john wrote to accountNames.txt: John  User john read accountNames.txt as:  John  User john logged out  Login failed: invalid username or password  Error: no user logged in  User admin logged in  Group Testers created  User john added to group Testers  File log.txt with owner admin and default permissions created  User admin wrote to log.txt: Today was a good day.  User rob created | Notify only on change  User rob added to group Testers  Group Programmers created  User veronica created  Notify only on change  User veronica added to group Programmers  User admin logged out  User veronica logged in  Always notify  File script.sh with owner veronica and default permissions created  User veronica wrote to script.sh: #!/bin/bash  User veronica wrote to script.sh: echo count to 4:  The ACL for script.sh appended by veronica to include Testers:X  User veronica logged out  User rob logged in  Never notify  User rob denied read access to script.sh  Program executed script.sh  User rob logged out  User veronica logged in  User veronica wrote to script.sh: COUNTER=0  User veronica wrote to script.sh: while [ $COUNTER -lt 5 ]; do  User veronica wrote to script.sh: echo $COUNTER  User veronica wrote to script.sh: let COUNTER=COUNTER+1 | User veronica wrote to script.sh: done  User veronica read script.sh as:  #!/bin/bash  echo count to 4:  COUNTER=0  while [ $COUNTER -lt 5 ]; do  echo $COUNTER  let COUNTER=COUNTER+1  done  User veronica wrote to script.sh: echo ...Done!  User veronica logged out  User john logged in  Program executed script.sh  User john logged out  User admin logged in  User admin wrote to accountNames.txt: Rob  User admin wrote to accountNames.txt: Veronica  User admin wrote to log.txt: Finished adding account names.  User admin read log.txt as:  Today was a good day.  Finished adding account names.  User admin read accountNames.txt as:  John  Rob  Veronica  User admin logged out |

accounts.txt 文件内容

admin h4cks\_\*&/ Always notify

john &DS4Hh Never notify

rob 237hd\_.L? Never notify

veronica 47\*3nd\_)/d\ Always notify

groups.txt 文件内容

Administrators: admin

Programmers: veronica

Testers: john rob

Users: john rob veronica

files.txt 文件内容

accountNames.txt john (john:F,Administrators:F)

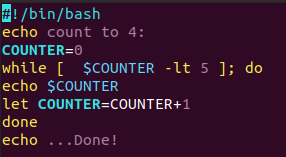
log.txt admin (admin:F,Administrators:F,Users:R)

script.sh veronica (veronica:F,Administrators:F,Testers:X)

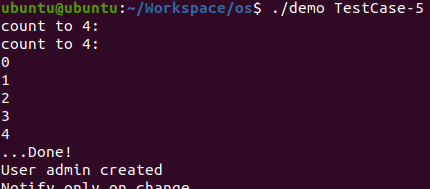
log.txt 文件内容



script.sh 文件内容



程序截图



通过比较TestCase-3的输入内容和audit.txt的输出内容，我们主要得出两点：

1. 当账号的UAC状态为Never时，执行program命令不需要输入yes；反之，当账号的UAC状态不为Never时，每条program命令需要输入yes来确认执行，否则不会执行program命令。
2. program execute命令通过调用system()函数来实现script.sh中的程序，并可以正确输出结果。