# Linux平台C程序基础

实验名称：Linux平台C程序基础

学时安排：2课时 指导老师：李赞

实验类别：验证型、设计型 实验要求：1人1组

学号： 姓名： 班级：

## 一、实验目的和任务

1. 本实验要求掌握C程序的基本编写和编译运行。

2. 本实验要求掌握Linux常用命令的使用。

## 二、实验设备介绍

1.软件需求： win10操作系统，VMware workstation，ubuntu18。

2.硬件需求: PC内存大于1G，硬盘空间20G。

## 三、注意事项和要求

1.背诵Linux平台常用的命令。

2.检查源程序是否输入正确，编译过程是否正确。

3.源代码应以UTF-8编码方式保存。

## 四、实验内容和步骤

### 4.1UBUNTU的启动与基础操作

#### 4.1.1 Linux中常用命令和操作

Linux操作系统平台提供了命令窗口和图形化窗口，命令窗口功能是很强大的，掌握Linux平台常用的命令是使用Linux的基础，虽然命令的数量非常多，但常用的命令约二十个左右。下表列出了Linux中较常用的一些命令及功能说明。

|  |  |
| --- | --- |
| 命令 |  |
| pwd | 显示用户当前完整的工作目录。 |
| su | 更改登录用户，默认为root(超级用户) |
| cd | 切换到指定目录，可使用tab键进行自动补全。 |
| cp | 将一个文件拷贝到另一个位置，原文件不变。 |
| ls | 显示当前目录下的文件，包括子目录。 |
| ls -l | 以详细信息显示当前目录下文件信息。 |
| gcc | gcc本身一个可编译C及C++代码的工具， |
| ifconfig | 查看和设置本机IP地址和掩码。 |
| ps -ef | 查看当前运行的所有进程。 |
| chmod | 修改文件或目录的权限  $ chmod u=rwx,g=rx,o=x file11  $ chmod 751 file 　　　 给file的属主分配读、写、执行(7)的权限，给file的所在组分配读、执行(5)的权限，给其他用户分配执行(1)的权限 |
| insmod | insmod命令用于将给定的模块加载到内核中。通常是设备驱动程序。 |
| uname | 用于打印当前系统（内核版本号、硬件架构、主机名称和操作系统类型等信息） |
| uname -r | 查看内核版本号 |
| uname -a | 显示当前操作系统内核版本号、硬件架构、主机名称和操作系统类型。 |
| lsmod | lsmod命令用于显示已经加载到内核中的模块的状态信息。 |
| man | 后面跟一个Linux中的命令的名字，系统给出这个命令的综述、对应参数的功能。用户在看完信息后要输入q ，退出man信息，返回到命令行状态。 |
| dmesg | 显示系统启动过程信息，该信息存储在kernel ring buffer中。 |
| cat | 显示文件全部内容。 |
| more | 查看文件命令，可翻页。 |
| nano | 文本行格式的编辑器。 |
| mount | 加载外部存储设备到系统文件系统，如加载:U盘、光盘、nfs等 |
| umount | 卸载通过mount命令的文件设备 |
| declare -x | 指定的变量会成为环境变量，可供shell以外的程序来使用； |

用mount命令可以加载U盘，光盘，nfs文件系统等可移动的存储设备到当前系统，这点与window的自动识别外部存储设备有一定区别，而且不用的外部存储设备还要使用umount命令卸载。用mount命令加载光驱设备到/mnt/cdrom下，接着cd命令 进入该目录，用ls列出光盘内容后，要换一张令外的光盘，则需使用umount命令卸载光盘设备。

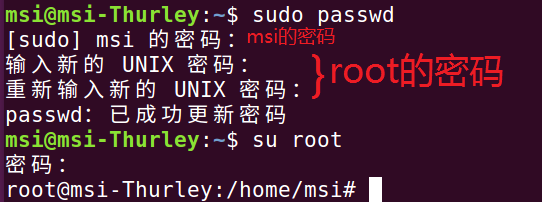
#### 4.1.2在Ubuntu下将普通用户提升到root权限

Ubuntu 18.04后默认是没有设置root密码的，使用下面的步骤给root账号设置密码。

1、用当前登录用户打开终端，在终端输入命令 sudo passwd，输入当前用户的密码然后回车

2、系统提示输入root新密码，输入密码确认。

3、系统提示再输入一次新密码以确认，给root账号设置密码成功。



#### **4.1.3 Ubuntu使用ifconfig和route命令设置网络地址**

Ubuntu中使用ifconfig命令查看或设置网卡信息，但是使用su命令时需要增加sudo，如下图：



如果ifconfig没有显示eth0网卡，请启用你的网卡：

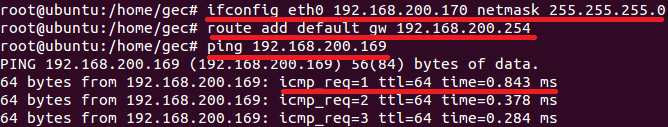


使用ifconfig命令设置网卡的IP地址和掩码

ifconfig eth0 172.16.56.106 netmask 255.255.255.0

下面命令用于设置默认路由。

route add default gw 172.16.56.254



### 4.2 UBUNTU平台的C程序编辑编译和运行

不同平台的C程序遵守的规范都是一样的，c程序包含两种后缀的文件, .c文件和.h文件，.c称作C源文件，文件内容是C代码，.h文件称作C头文件，里面也是c源代码。 在整个项目中，编译器管理一个项目仅有一个main函数。在不同平台编译C程序的工具是不同的，VisualStudio工具是Windows平台的编程工具，并不能在Linux平台上使用。Linux平台也有多个可以编译C语言的工具，例如Linux平台名气比较大的是GCC编译器，一般还有g++和make工具，在ubuntu中可使用命令安装这一组编译相关工具:

sudo apt-get install build-essential

GCC是Linux操作系统下一个非常重要的源代码编译工具,有着许多重要的选项,支持许多不同语言的编译,如C、C++、Ada、Fortran、Objective、Perl、Python、Ruby以及Java等,甚至Linux的许多内核和许多其他自由软件以及开放源码应用程序都是用C语言编写并经gcc编译而成的。

GCC是一个以命令行形式操作的编译工具。

编译命令：gcc hello.c -o hello

执行生成的可执行文件：./hello

下表列出GCC工具的常用参数开关和作用。

|  |  |
| --- | --- |
| 参数开关 | 作用 |
| o | 指定生成的输出文件； |
| E | 仅执行编译预处理； |
| S | 将C代码转换为汇编代码； |
| wall | 显示警告信息； |
| c | 仅执行编译操作，不进行连接操作。 |
| g | 为gdb调试工具工具产生符号信息。调试程序用。 |
| v | gcc执行时执行的详细过程，gcc及其相关程序的版本号 |
| x | 指定文件使用的编程语言，如gcc -x c hello.pig |
| lpthread | 支持线程代码 |

实际上GCC工具的参数有很多个，有些参数特别罕用，以下网址供参考。

http://www.cnblogs.com/zhangsir6/articles/2956798.html

Linux平台编辑源代码的工具也很多，例如最原始的VIM工具使用下面的命令进入编辑hello1.c源代码。

$vi hello1.c 进入VIM

还有nano文本编辑工具，以及图形化的gedit代码编辑工具。

如果在windows平台使用虚拟机的环境下，也可以在windows平台编辑好代码(保存为utf8编码)，然后将代码文件拖放入虚拟机中。目前比较好用的是code工具，它实际上是vscode的linux版，使用起来非常方便，推荐本门课程使用code工具编写源码。

在linux平台中使用gcc工具编译C程序，命令形式如下：

gcc -o ff hello.c 运行编译后的程序命令为 ./ff

#### 4.2.1指针与数组在c程序中的区别

请运行下面的代码查看指针与数组在C程序中的区别。

//001.c 指针与数组对于sizeof运算的区别

#include <stdio.h>

#include <stdlib.h>

int main()

{

char \*p1 =malloc(100);

char p2[100];

printf("sizeof p1 = %d , sizeof p2 = %d\n",sizeof(p1),sizeof(p2));

free(p1);

return 0;

}

#### 4.2.2使用指针作为函数参数

下面是一个简单的C程序，该程序中test函数没有实现对bb变量修改的效果。如果要在函数中达到对参数变量值修改可使用传地址方式的参数引用。

//002.c 函数参数的传值方式

#include <stdio.h>

void test(int i)

{

int aa = 0;

int dd = 0;

i =10;

}

int main(void)

{

int bb = 1;

int cc = 99;

test(bb);

printf("bb = %d\n", bb);

return 0;

}

//函数传参的三种方式：传值、传址、传引用

#include<stdio.h>

void change(int\*a, int&b, int c)

{

c=\*a;

b=30;

\*a=20;

}

int main ( )

{

int a=10, b=20, c=30;

change(&a,b,c);

printf(“%d,%d,%d,”,a,b,c)；

return 0；

}

结果：20 30 30

解析：

该题考察函数传参问题。

1，指针传参 -> 将变量的地址直接传入函数，函数中可以对其值进行修改。（传址）

2，引用传参 -> 将变量的引用传入函数，效果和指针相同，同样函数中可以对其值进行修改。（传引用）

3，值传参 -> 在传参过程中，首先将c的值复制给函数c变量，然后在函数中修改的即是函数的c变量，然后函数返回时，系统自动释放变量c。而对main函数的c没有影响。（传值）

修改下面的swap函数，使得swap函数能够实现对两个变量值的交换。

//003.c 函数参数的传地址方式

#include <stdio.h>

void swap(int \*i)

{

int aa = 0;

int dd = 0;

(\*i) =10;

}

int main(void)

{

int bb = 1;

int cc = 99;

swap (&bb);

printf("bb = %d\n", bb);

return 0;

}

#### 4.2.3 字符串反转操作

下面是一段C程序，该程序原本要实现对一给定字符串的逆序，该程序存在较多的问题，请一一调试并排除错误，使其能输出正确逆序的字符串。

//004.c 实现对字符串的逆序运算

#include <stdio.h>

int display1(char \*string);

int reverse(char \*string);

int main()

{

char string[] = "Embeded Linux in ZhongShan Institute";

char \*ptr = NULL;

display1(string);

ptr = reverse(string);

printf("the string len is %d\n",strlen(ptr));

}

int display1(char \* string)

{

printf("The original string is %s\n",string);

}

int reverse(char \* string1)

{

char \*string2;

int size = 0;

int i = 0;

size = strlen(string1);

string2 = (char \*)malloc(size + 1);

for(i = 0; i < size; i++)

{

string2[size – i] = string1[i];

}

string2[size+1] = ' ';

printf("The string afterward is %s\n",string2);

}

程序作业：1.请指出原程序的错误原因，2.请给出程序正确的运行截图。

#### 4.2.4 Main函数参数argc，argv的用法

main函数是C程序的入口函数，带有参数argc，argv的main函数声明如下：

int main(int argc, char\*\* argv)

argc 是指命令行输入参数的个数(以空白符分隔) argv是字符串数组，其内容包含了命令行的所有字符串组成的参数。在Debian或者ubuntu系统中调试下面的示例代码：

//005.c main函数的参数

#include "stdio.h"

int main(int argc, char\*\* argv)

{

printf("argc = %d\n",argc);

printf("argv[0] = %s\n",argv[0]);

printf("argv[1] = %s\n",argv[1]);

}

使用下面的命令编译源代码：

gcc -o myprogram 005.c

在终端执行命令：./myprogram 192.168.0.1，试观察程序结果，体会argc,argv[0],argv[1]分别代表的意思。更换命令，观察结果有何不同./myprogram 及 ./myprogram 123 456。在后续的网络通讯程序中，机器的IP地址将以命令行参数的方式读入程序。要注意的是argv[]是char \*类型，需要进一步类型转换才能使用，例如端口号。

#### 4.2.5 C程序中全局变量与局部变量

C程序中全局变量在程序所有地方都可访问，而局部变量仅限于其声明的作用域，例如在函数内部声明的变量，即使系统事先给其分配了内存，也不可以通过传递地址的方式在函数以外使用。例如下面的代码：

//006.c 全局变量与局部变量

#include "stdio.h"

int \*p = NULL;

int x;

int \*fFun(void)

{

int i = 0;

printf("variant i's address is 0x%x and value is %d\n",&i, i);

return &i;

//return &x;

}

void subFun(void)

{

(\*p)--;

printf("pointer p's address is 0x%x and value is %d\n",p, \*p);

}

void gFun(void)

{

int j;

printf("variant j's address is 0x%x and value is %d\n",&j, j);

for(j = 0;j<10;j++)

{

subFun();

printf("%d\n",j);

}

}

int main(void)

{

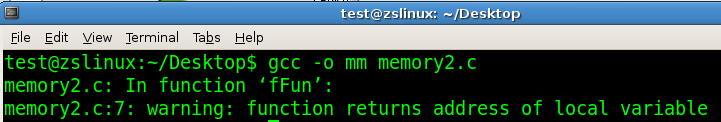
p = fFun();

gFun();

}

除了用malloc函数返回的指针地址外，所有的局部变量都不可以在函数外使用。也不可以将局部变量地址return出去。

该程序的fFun函数返回的值是局部变量的地址，编译会给出异常提示，运行程序结果不可预计。



将局部变量声明为全局变量后，fFun函数不再提示编译异常。

#### 4.2.6 C程序中实现字符串拷贝

c语言中的字符串是以'\0'结尾的连续的字符，下面是一个实现将源字符串内容拷贝到目标字符串的函数实现。

//007.c 字符串的拷贝

#include <stdio.h>

#include <memory.h>

#define MAX\_MSG\_LEN 1024

void copystr(char\* dest, const char \*src)

{

//dest是目标字符串指针首地址，src是源字符串指针首地址

//将src指向的字符串内容复制到由dest指向的地址中

if(dest!=NULL && src!=NULL)

{

char \*strD=dest;

while ((\*strD++=\*src++)!='\0');

}

}

int main(int argc, char \*\*argv)

{

char \*p = "This is an exam for Linux Software. Please Coding carefully and check slowly";

char szMsg[MAX\_MSG\_LEN];

memset(szMsg, 0 ,sizeof(szMsg));

copystr(szMsg, p);

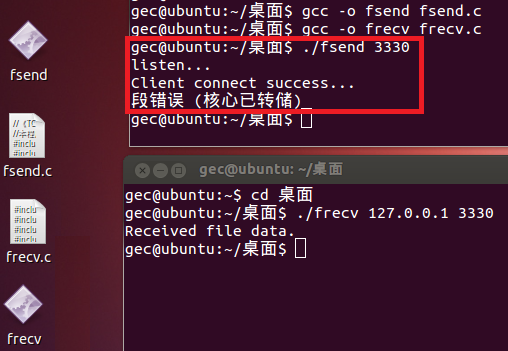
printf("The string is %s\n", szMsg);

}

## 五、程序调试中遇到的问题和解决过程及运行结果

1.将上述每个程序先使用gcc 编译并运行，请将程序结果进行截图。

2.分析过程中遇到问题，排查程序出错的原因，并将每个程序结果截图保存，例如：



排查程序出错原因，描述解决过程。