## task1:

#include <stdio.h>

#include <sys/stat.h>

#include <unistd.h>

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

{

if(argc < 2)

{

printf("ERROR:Please input the file name.\n");

return 0;

}

struct stat buf;

char \*path = argv[1];

int ret;

FILE \*fp=fopen(argv[1],"w+");

if(fp==NULL)

{

printf("open error\n");

return -1;

}

printf("creat success\n");

fclose(fp);

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

else

{

if(buf.st\_mode & S\_IRUSR)

printf("user read\n");

if(buf.st\_mode & S\_IWUSR)

printf("user write\n");

if(buf.st\_mode & S\_IXUSR)

printf("user exe\n");

if(buf.st\_mode & S\_IRGRP)

printf("GRP read\n");

if(buf.st\_mode & S\_IWGRP)

printf("GRP write\n");

if(buf.st\_mode & S\_IXGRP)

printf("GRP exe\n");

if(buf.st\_mode & S\_IROTH)

printf("OTH read\n");

if(buf.st\_mode & S\_IWOTH)

printf("OTH write\n");

if(buf.st\_mode & S\_IXOTH)

printf("OTH exe\n");

printf("=============================================\n"); // 以rwx方式表示

printf("The file authority is: ");

int i;

for(i = 2; i >= 0; i--)

{

if(buf.st\_mode & 1 << (i \* 3 + 2))

printf("r");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 1))

printf("w");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 0))

printf("x");

else

printf("-");

}

printf("\n");

printf("=============================================\n"); // 以数字方式表示

printf("The file authority is: 0%o\n",buf.st\_mode & (S\_IRUSR | S\_IWUSR | S\_IXUSR | S\_IRGRP | S\_IWGRP | S\_IXGRP | S\_IROTH | S\_IWOTH | S\_IXOTH));

}

return 0;

}

## task2:

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <string.h>

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

{

if(argc < 3)

{

printf("ERROR:Please input the file name.\n");

return 0;

}

struct stat buf;

char \*path = argv[1];

int ret;

char \*pnewname="testfile\_chg";

FILE \*fp;

fp=fopen(argv[1],"a+");

if(fp==NULL)

{

printf("fopen error\n");

}

if((int)fwrite(argv[2],1,(size\_t)strlen(argv[2]),fp)<0)

{

printf("write error\n");

return -1;

}

printf("write over\n");

fclose(fp);

printf("chang file name over\n");

rename(path,pnewname);

if(ret = stat(pnewname, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

buf.st\_mode &= ~S\_IWGRP;

buf.st\_mode &= ~S\_IWOTH;

if(ret = chmod(pnewname, buf.st\_mode) != 0)

{

printf("ERROR: execute chmod function error.\n");

return 0;

}

int i;

for(i = 2; i >= 0; i--)

{

if(buf.st\_mode & 1 << (i \* 3 + 2))

printf("r");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 1))

printf("w");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 0))

printf("x");

else

printf("-");

}

printf("\n");

return 0;

}

## task3:

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <stdlib.h>

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

{

struct stat buf;

char \*path = argv[1];

int ret;

FILE \*fp;

char szread[100]={0};

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

if(argc < 2)

{

printf("ERROR:Please input the file name.\n");

return 0;

}

fp=fopen(path,"r");

if(fp==NULL)

{

printf("open error\n");

return -1;

}

if(fread(szread,1,sizeof(szread),fp)<=0)

{

printf("read error\n");

return -1;

}

printf("file content is %s\n",szread);

fclose(fp);

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

// buf.st\_mode &= 0;

// buf.st\_mode = S\_IRUSR | S\_IWUSR | S\_IRGRP | S\_IROTH;

buf.st\_mode=S\_IWUSR;

if(ret = chmod(path, buf.st\_mode) != 0)

{

printf("ERROR: execute chmod function error.\n");

return 0;

}

printf("The file authority is: ");

int i;

for(i = 2; i >= 0; i--)

{

if(buf.st\_mode & 1 << (i \* 3 + 2))

printf("r");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 1))

printf("w");

else

printf("-");

if(buf.st\_mode & 1 << (i \* 3 + 0))

printf("x");

else

printf("-");

}

printf("\n");

return 0;

}

## task4:

#include <stdio.h>

#include <stdlib.h>

#include <sys/stat.h>

#include <time.h>

#include <string.h>

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

{

struct stat buf;

char \*pnewfile="testfile\_bak";

char szbuf[100]={0};

FILE \*fpold=NULL,\*fpnew=NULL;

int len=0;

if(argc <2)

{

printf("arg error\n");

}

fpold=fopen(argv[1],"r");

if(stat(argv[1],&buf)!=0)

{

printf("stat error\n");

}

printf("user gid is %d,grp id is %d\n",buf.st\_uid,buf.st\_gid);

printf("visi time is %ld,last time is %ldo\n",buf.st\_atime,buf.st\_mtime);

printf("vist time is %s,last time is %s\n",ctime((const time\_t \*)&buf.st\_atime),ctime((const time\_t \*)&buf.st\_mtime));

if(fpold==NULL)

{

printf("open file %s error\n",argv[1]);

return -1;

}

if((fpnew=fopen(pnewfile,"w+"))==NULL)

{

printf("open file %s error\n",pnewfile);

return -1;

}

if((len=fread(szbuf,1,sizeof(szbuf),fpold))<0)

{

printf("fread error\n");

return -1;

}

if(fwrite(szbuf,1,len,fpnew)<0)

{

printf("write error\n");

return -1;

}

printf("rewrite over\n");

fclose(fpold);

fclose(fpnew);

if( remove(argv[1])==0)

{

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

}

return 0;

}

## task5:

/\* 将/etc/passwd 的所有者和组都设为root \*/

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

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

{

if(argc < 2)

{

printf("ERROR:Please input the file name.\n");

return 0;

}

struct stat buf;

char \*path = argv[1];

int ret;

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

printf("Group ID before change is: %u\n", buf.st\_gid);

if(ret = chown(path, -1, 0) != 0)

{

printf("ERROR: execute chown function error.\n");

return 0;

}

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

printf("Group ID after change is: %u\n", buf.st\_gid);

return 0;

}

## task6:

#include <stdio.h>

#include <sys/stat.h>

#include <utime.h>

#include <time.h>

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

{

if(argc < 2)

{

printf("ERROR:Please input the file name.\n");

return 0;

}

struct stat buf;

char \*path = argv[1];

int ret;

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

printf("last access time before change is: %s\n", ctime(&(buf.st\_atime)));

printf("last modify time before change is: %s\n", ctime(&(buf.st\_mtime)));

if(ret = utime(path, NULL) != 0)

{

printf("ERROR: execute utime function error.\n");

return 0;

}

if(ret = stat(path, &buf) != 0)

{

printf("ERROR: execute stat function error.\n");

return 0;

}

printf("last access time after change is: %s\n", ctime(&(buf.st\_atime)));

printf("last modify time after change is: %s\n", ctime(&(buf.st\_mtime)));

return 0;

}

## //task7:

#include <stdio.h>

#include <sys/select.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

int main()

{

int keyboard;

int ret;

fd\_set readfds;

char key;

struct timeval timeout;

char \*path = "/dev/tty";

keyboard = open(path, O\_RDONLY | O\_NONBLOCK); //打开键盘文件

if(keyboard < 0)

{

printf("open error!\n");

return 1;

}

printf("keyboard is %d\n", keyboard);

timeout.tv\_sec=0; //无阻塞

timeout.tv\_usec=0;

//无阻塞检测输入

while(1)

{

FD\_ZERO(&readfds); //每次循环都要清空集合，select检测描述文件中是否有可读的，从而能检测描述符变化

FD\_SET(keyboard, &readfds);

ret = select(keyboard+1, &readfds, NULL, NULL, &timeout); //select检测描述文件中是否有可读的

if(ret < 0)

{

printf("select error!\n");

return 1;

}

//检测读文件描述符集合，一直在循环，监视描述符的变化

ret = FD\_ISSET(keyboard, &readfds);

if(ret > 0)

{

read(keyboard, &key, 1);

if('\n' == key)

continue;

printf("the input is %c\n", key);

if ('q' == key)

break;

}

}

return 0;

}