Name: Kaustav Ghosh

Reg no: 180905188

Roll no: 29

Date:11/12/2020

Q1

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <stdio.h>

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

{

struct stat sb;

int ret;

if (argc < 2)

{

fprintf(stderr, "usage: %s <file> \n", argv[0]);

return 1;

}

ret = stat(argv[1], &sb);

if (ret)

{

perror("stat");

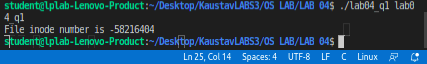
return 1;

}

printf("File inode number is %d \n", argv[1], sb.st\_ino);

return 0;

}



Q2

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <stdio.h>

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

{

struct stat sb;

int ret;

if (argc < 2)

{

fprintf(stderr, "usage: %s <file> \n", argv[0]);

return 1;

}

ret = stat(argv[1], &sb);

if (ret)

{

perror("stat");

return 1;

}

printf(" ID of device containing file %d \n", sb.st\_dev);

printf(" inode number %d \n", sb.st\_ino);

printf(" permissions %d \n", sb.st\_mode);

printf(" number of hard links %d \n", sb.st\_nlink);

printf(" user ID of owner %d \n", sb.st\_uid);

printf(" group ID of owner %d \n", sb.st\_gid);

printf(" device ID (if special file) %d \n", sb.st\_rdev);

printf(" total size in bytes %d \n", sb.st\_size);

printf(" blocksize for filesystem I/O %d \n", sb.st\_blksize);

printf(" number of blocks allocated %d \n", sb.st\_blocks);

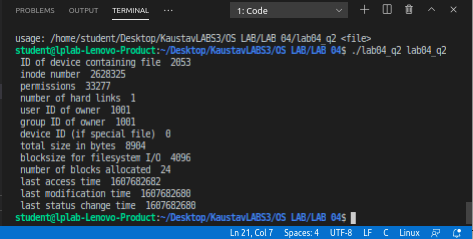
printf(" last access time %d \n", sb.st\_atime);

printf(" last modification time %d \n", sb.st\_mtime);

printf(" last status change time %d \n", sb.st\_ctime);

return 0;

}



Q3

#define \_POSIX\_SOURCE

#include <fcntl.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

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

{

struct stat sb;

int ret;

if (argc < 2)

{

fprintf(stderr, "usage: %s <file> \n", argv[0]);

return 1;

}

char sln[] = "test.link";

puts("Before link()");

system("readlink -f test.\*");

// Hard link

if (link(argv[1], sln) != 0)

{

perror("link error");

unlink(argv[1]);

}

else

{

puts("After link()");

system("readlink -f test.\*");

unlink(argv[1]);

puts("after first unlink()");

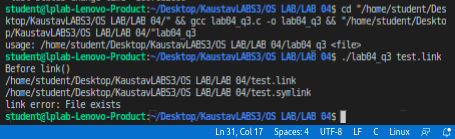
system("readlink -f test.\*");

unlink(sln);

}

return 0;

}



Q4

#define \_POSIX\_SOURCE

#include <fcntl.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

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

{

struct stat sb;

int ret;

if (argc < 2)

{

fprintf(stderr, "usage: %s <file> \n", argv[0]);

return 1;

}

char sln[] = "test.symlink";

puts("Before symlink()");

system("readlink -f test.\*");

// Soft Link

if (symlink(argv[1], sln) != 0)

{

perror("symlink error");

unlink(argv[1]);

}

else

{

puts("After symlink()");

system("readlink -f test.\*");

unlink(argv[1]);

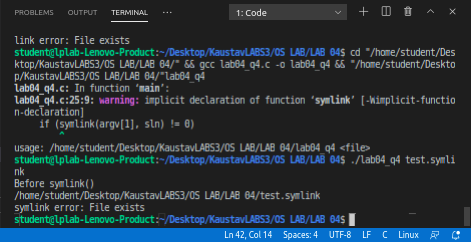
puts("after first unlink()");

system("readlink -f test.\*");

unlink(sln);

}

return 0;

}