

Lab 4 OS

Name: Paawan Kohli

Reg No: 180905416

Roll No: 52

Q1. Write a program to find the inode number of an existing file in a directory. Take the input as a filename and print the inode number of the file.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/stat.h>

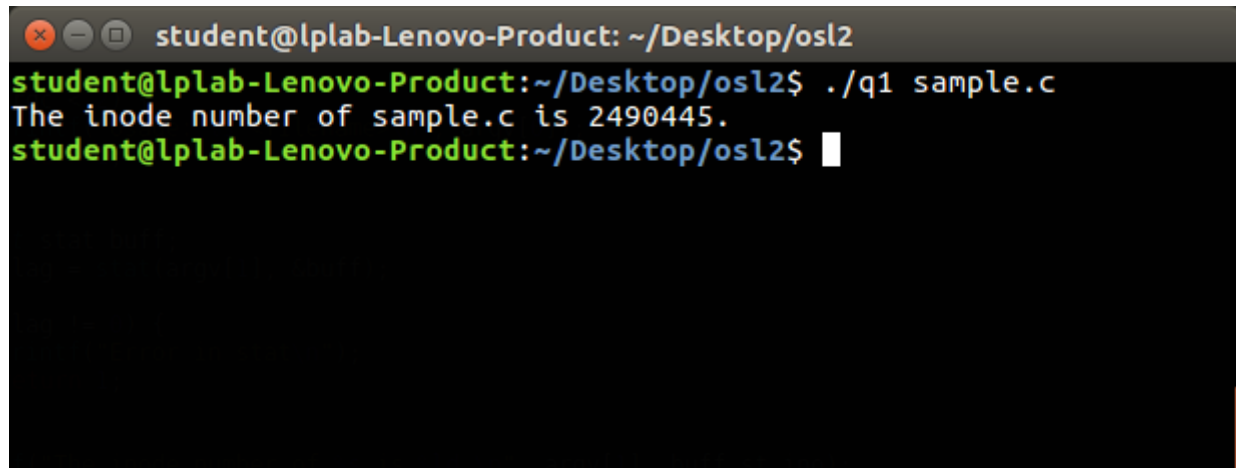
int main (int argc, char *argv[]) {

    if (argc < 2) {
        printf("Usage: %s <filename>\n", argv[0]);
        return 1;
    }

    struct stat buff;
    int flag = stat(argv[1], &buff);

    if (flag != 0) {
        printf("Error in stat\n");
        return 1;
    }

    printf("The inode number of %s is %ld.\n", argv[1], buff.st_ino);
    return 0;
}
```

A terminal window with a dark background and light green text. The title bar shows 'student@lplab-Lenovo-Product: ~/Desktop/osl2'. The prompt is 'student@lplab-Lenovo-Product:~/Desktop/osl2\$'. The user enters './q1 sample.c'. The output is 'The inode number of sample.c is 2490445.'. The prompt returns to 'student@lplab-Lenovo-Product:~/Desktop/osl2\$' with a cursor.

```
student@lplab-Lenovo-Product: ~/Desktop/osl2
student@lplab-Lenovo-Product:~/Desktop/osl2$ ./q1 sample.c
The inode number of sample.c is 2490445.
student@lplab-Lenovo-Product:~/Desktop/osl2$
```

Q2. Write a program to print out the complete stat structure of a file.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/stat.h>
#include <time.h>

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

    if (argc < 2) {
        printf("Usage: %s <filename>\n", argv[0]);
        return 1;
    }

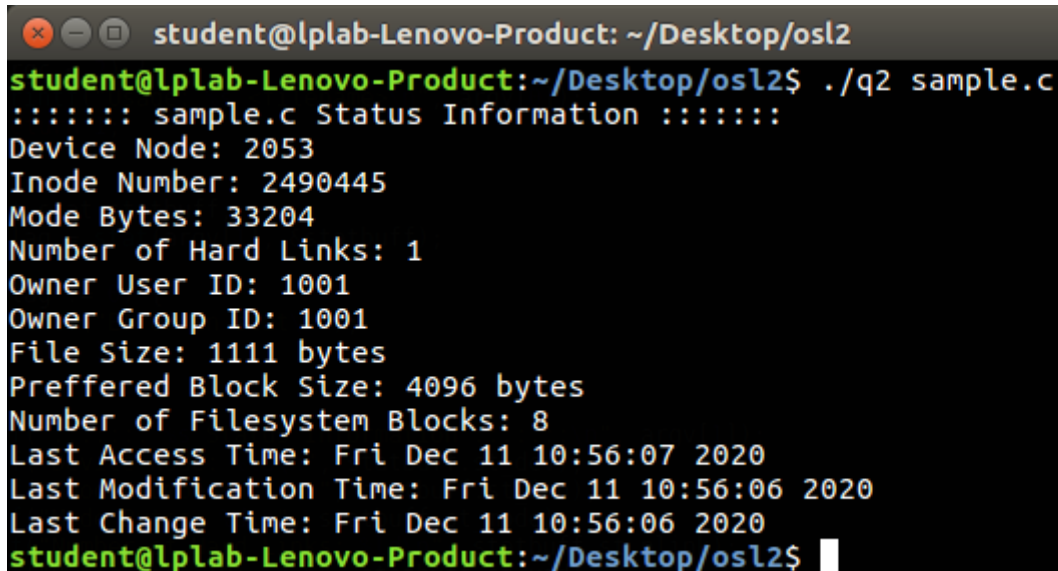
    struct stat statbuff;
    int flag = stat(argv[1], &statbuff);

    if (flag != 0) {
        printf("Error in stat\n");
        return 1;
    }

    printf("::::::::: %s Status Information ::::::::::\n", argv[1]);
    printf("Device Node: %ld\n", statbuff.st_dev);
    printf("Inode Number: %ld\n", statbuff.st_ino);
    printf("Mode Bytes: %d\n", statbuff.st_mode);
    printf("Number of Hard Links: %ld\n", statbuff.st_nlink);
    printf("Owner User ID: %d\n", statbuff.st_uid);
    printf("Owner Group ID: %d\n", statbuff.st_gid);
    printf("File Size: %ld bytes\n", statbuff.st_size);
    printf("Preffered Block Size: %ld bytes\n", statbuff.st_blksize);
    printf("Number of Filesystem Blocks: %ld\n", statbuff.st_blocks);

    // char* ctime(time_t*)
    printf("Last Access Time: %s", ctime(&statbuff.st_atime));
    printf("Last Modification Time: %s", ctime(&statbuff.st_mtime));
    printf("Last Change Time: %s", ctime(&statbuff.st_mtime));

    return 0;
}
```

A terminal window with a dark background and light-colored text. The window title is "student@lplab-Lenovo-Product: ~/Desktop/osl2". The prompt is "student@lplab-Lenovo-Product:~/Desktop/osl2\$". The user has entered the command ". /q2 sample.c". The output of the program is displayed, showing status information for the file "sample.c". The output includes fields like Device Node, Inode Number, Mode Bytes, Number of Hard Links, Owner User ID, Owner Group ID, File Size, Preffered Block Size, Number of Filesystem Blocks, and timestamps for Last Access, Last Modification, and Last Change. The prompt is now "student@lplab-Lenovo-Product:~/Desktop/osl2\$" with a cursor.

```
student@lplab-Lenovo-Product: ~/Desktop/osl2
student@lplab-Lenovo-Product:~/Desktop/osl2$ ./q2 sample.c
::::::::: sample.c Status Information ::::::::::
Device Node: 2053
Inode Number: 2490445
Mode Bytes: 33204
Number of Hard Links: 1
Owner User ID: 1001
Owner Group ID: 1001
File Size: 1111 bytes
Preffered Block Size: 4096 bytes
Number of Filesystem Blocks: 8
Last Access Time: Fri Dec 11 10:56:07 2020
Last Modification Time: Fri Dec 11 10:56:06 2020
Last Change Time: Fri Dec 11 10:56:06 2020
student@lplab-Lenovo-Product:~/Desktop/osl2$
```

Q3. Write a program to create a new hard link to an existing file and unlink the same. Accept the old path as input and print the newpath.

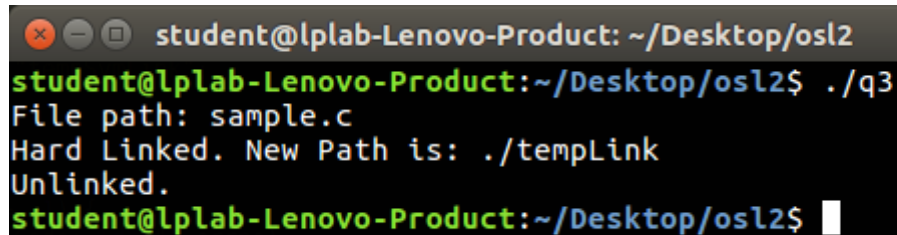
```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>

void main() {
    char oldPath[50], newPath[50] = "./tempLink";

    printf("File path: ");
    scanf("%s", oldPath);

    if (link(oldPath, newPath) == -1) {
        printf("Hard Linking error. Code: %d\n", errno);
        exit(1);
    } else {
        printf("Hard Linked. New Path is: %s\n", newPath);
    }

    if (unlink(newPath) == -1) {
        printf("Unlinking error. Code: %d\n", errno);
        exit(1);
    } else {
        printf("Unlinked.\n");
    }
}
```

A terminal window with a dark background and light-colored text. The title bar shows a red close button, a yellow minimize button, and a green maximize button, followed by the text 'student@lplab-Lenovo-Product: ~/Desktop/osl2'. The terminal content shows the execution of a program: the prompt 'student@lplab-Lenovo-Product:~/Desktop/osl2\$' is followed by the command './q3'. The program outputs 'File path: sample.c', 'Hard Linked. New Path is: ./tempLink', and 'Unlinked.' before returning to the prompt 'student@lplab-Lenovo-Product:~/Desktop/osl2\$' with a cursor.

```
student@lplab-Lenovo-Product: ~/Desktop/osl2
student@lplab-Lenovo-Product:~/Desktop/osl2$ ./q3
File path: sample.c
Hard Linked. New Path is: ./tempLink
Unlinked.
student@lplab-Lenovo-Product:~/Desktop/osl2$
```

Q4. Write a program to create a new soft link to an existing file and unlink the same. Accept the old path as input and print the newpath.

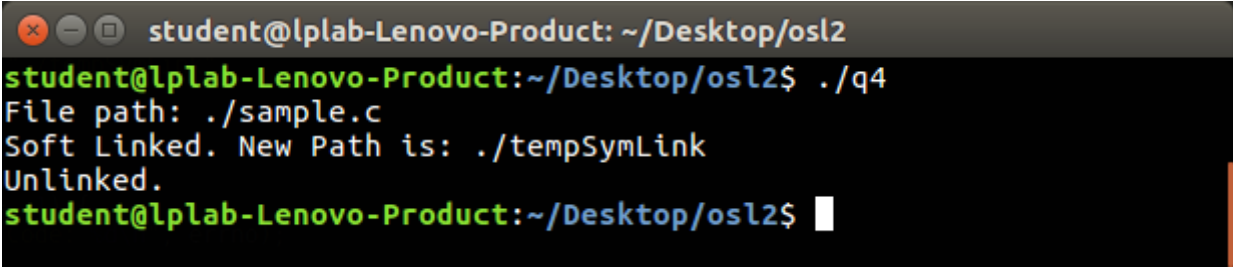
```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>

void main() {
    char oldPath[50], newPath[50] = "./tempSymLink";

    printf("File path: ");
    scanf("%s", oldPath);

    if (symlink(oldPath, newPath) == -1) {
        printf("Soft Linking error. Code: %d\n", errno);
        exit(1);
    } else {
        printf("Soft Linked. New Path is: %s\n", newPath);
    }

    if (unlink(newPath) == -1) {
        printf("Unlinking error. Code: %d\n", errno);
        exit(1);
    } else {
        printf("Unlinked.\n");
    }
}
```

A terminal window with a dark background and light green text. The title bar shows 'student@lplab-Lenovo-Product: ~/Desktop/osl2'. The prompt is 'student@lplab-Lenovo-Product:~/Desktop/osl2\$'. The user enters './q4'. The output is 'File path: ./sample.c', 'Soft Linked. New Path is: ./tempSymLink', and 'Unlinked.'. The prompt returns.

```
student@lplab-Lenovo-Product: ~/Desktop/osl2
student@lplab-Lenovo-Product:~/Desktop/osl2$ ./q4
File path: ./sample.c
Soft Linked. New Path is: ./tempSymLink
Unlinked.
student@lplab-Lenovo-Product:~/Desktop/osl2$
```