**3. File system Internals**

3.3 Write a program to use link/unlink system call for creating logical link and identifying the difference using stat.

**Objectives:**

1. To learn about File system Internals.

**Theory:**

1) Link:

Name:

link - make a new name for a file

Syntax :

#include <unistd.h>

int link(const char \*oldpath, const char \*newpath);

Description:

* link() creates a new link (also known as a hard link) to an existing file.
* If newpath exists it will not be overwritten.
* This new name may be used exactly as the old one for any operation; both names refer to the same file (and so have the same permissions and ownership) and it is impossible to tell which name was the `original‟.

Return Value:

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

2) Unlink:

Name:

unlink - delete a name and possibly the file it refers to

Syntax :

#include <unistd.h>

int unlink(const char \*pathname);

Description:

* unlink() deletes a name from the filesystem. If that name was last link to a file and no processes have the file open the file is deleted and the space it was using is made available for reuse.
* If the name was the last link to a file but any processes still have the file open the file will remain in existence until the last file descriptor referring to it is closed.
* If the name referred to a symbolic link the link is removed. If the name referred to a socket, fifo or device the name for it is removed but processes which have the object open may continue to use it.

Return Value:

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

**Data Dictionary:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Variable/Function** | **Data type** | **Use** |
| 1 | old | char[] | Old pathname. |
| 2 | new | char[] | New pathname. |
| 3 | ch | char | Choice asking to unlink or not. |

**Program:**

#include<stdio.h>

#include<unistd.h>

int main()

{

char old[100];

char new[100];

char ch;

printf("Enter the old and new pathname: \n");

gets(old);

gets(new);

int n = link(old,new);

if(n==0)

{

printf("Linked successfully\n");

}

else

{

printf("Linked unsuccessfully\n");

}

printf("Do you want to unlink the new file?\n1:Y\n2:N\n");

scanf("%c",&ch);

if(ch=='Y'||ch=='y')

{

int m = unlink(new);

if(m==0)

{

printf("Unlinked successfully\n");

}

else

{

printf("Unlinked unsuccessfully\n");

}

}

else

{

printf("Not Unlinked\n");

}

}

**Output:**

it@it-OptiPlex-3046:~/Mayur/UOS$ gcc 3A\_c.c

it@it-OptiPlex-3046:~/Mayur/UOS$ ./a.out

Enter the old and new pathname:

/home/it/Mayur/UOS/Demo.txt

/home/it/Mayur/UOS/Demo1.txt Linked

successfully

Do you want to unlink the new file?

1:Y

2:N

Y

Unlinked successfully

[1]+ Killed

gedit 3B\_b.c

**Conclusion:**

1.The concepts of creating link or shortcut to file

2.Unlinking it understood through link unlink function calls.

3.The change in number of links takes place as we implement the program.

**References:**

[1] www.tutorialspoint.com/unix\_system\_calls