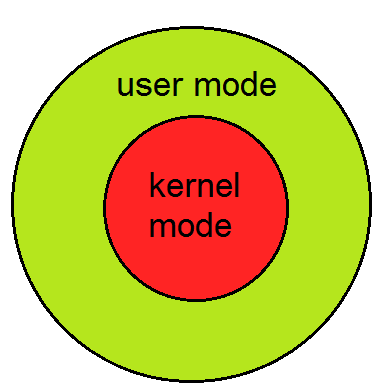
**SYSTEM CALLS**

System calls provide an interface to the services made available by an operating system. In other words, a **system call** is a mechanism that provides the interface between a process and kernel the operating system. These calls are generally available as routines written in C, C++ or assembly-language to access resources. For a simple command, the operating system executes thousands of system calls.

To understand system calls, first one needs to understand the difference between kernel mode and user mode of a CPU. There are two modes in which a program can execute are known as the kernel mode and user mode. Every modern operating system supports these two modes.



**Kernel Mode:**

* When CPU is in kernel mode, the code being executed can access any memory address and any hardware resource.
* Hence kernel mode is a very privileged and powerful mode.
* If a program crashes in kernel mode, the entire system will be halted.

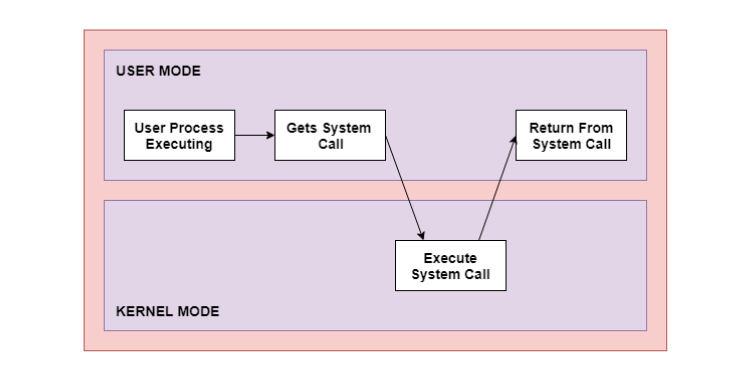
**User Mode:**

* When CPU is in user mode, the programs don't have direct access to memory and hardware resources.
* In user mode, if any program crashes, only that particular program is halted.
* That means the system will be in a safe state even if a program in user mode crashes.
* Hence, most programs in an OS run in user mode.

**Working of System call:**

When a program in user mode requires access to RAM or a hardware resource, it must ask the kernel to provide access to that resource through **system call**. When a program makes a system call, the mode is switched from user mode to kernel mode. This is called a **context switch**. Then the kernel provides the resource which the program requested. After that, another context switch happens which results in change of mode from kernel mode back to user mode.

Following figure representing the execution of the system call is given as follows-



Following are the steps on how a System Call works:

**Step 1:** The processor executes a process in the user mode until a system call interrupts it.  
**Step 2:** Then on a priority basis, the system call is executed in the kernel mode.  
**Step 3:** After the completion of system call execution, control returns to user mode.,  
**Step 4:** The execution resumes in Kernel mode.

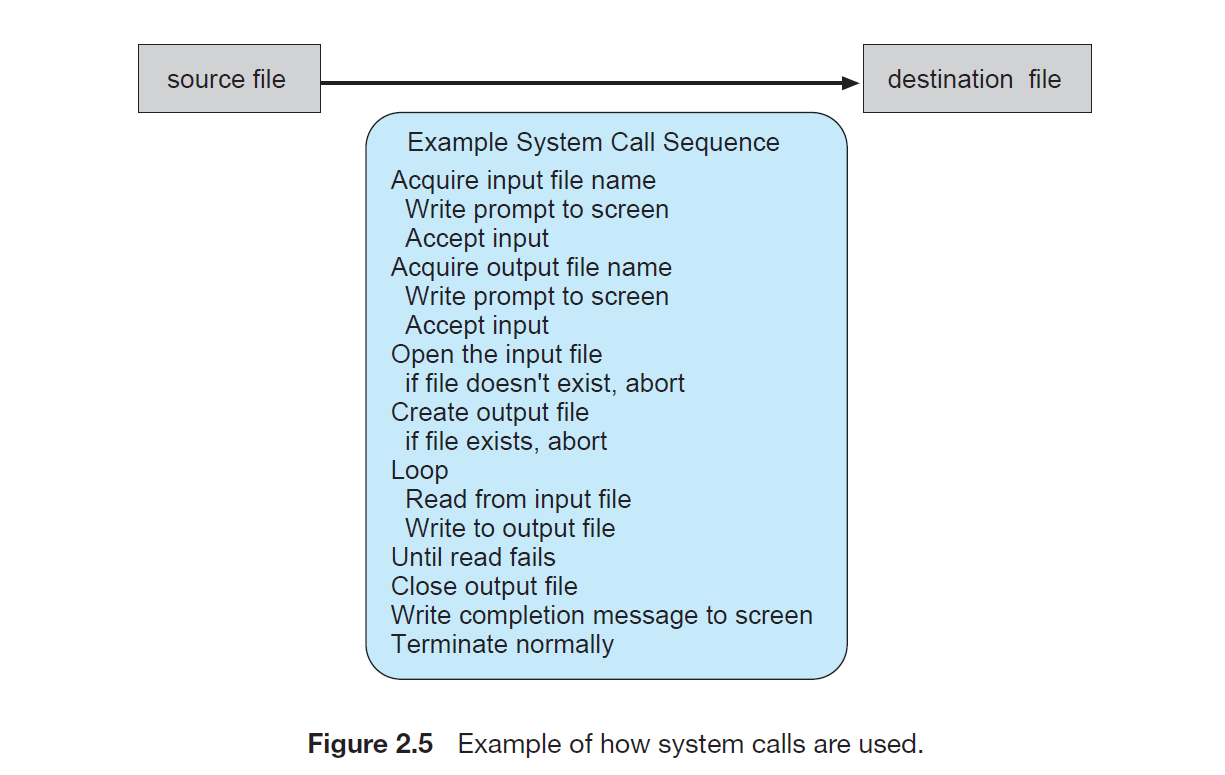
### Need for System Calls:

Following are the reasons we need system calls

* Reading and writing from files demand system calls.
* If a file system wants to create or delete files, system calls are required.
* System calls are used for the creation and management of new processes.
* Network connections need system calls for sending and receiving packets.
* Access to hardware devices like scanner, printer, need a system call.

## **Example of System Call:**

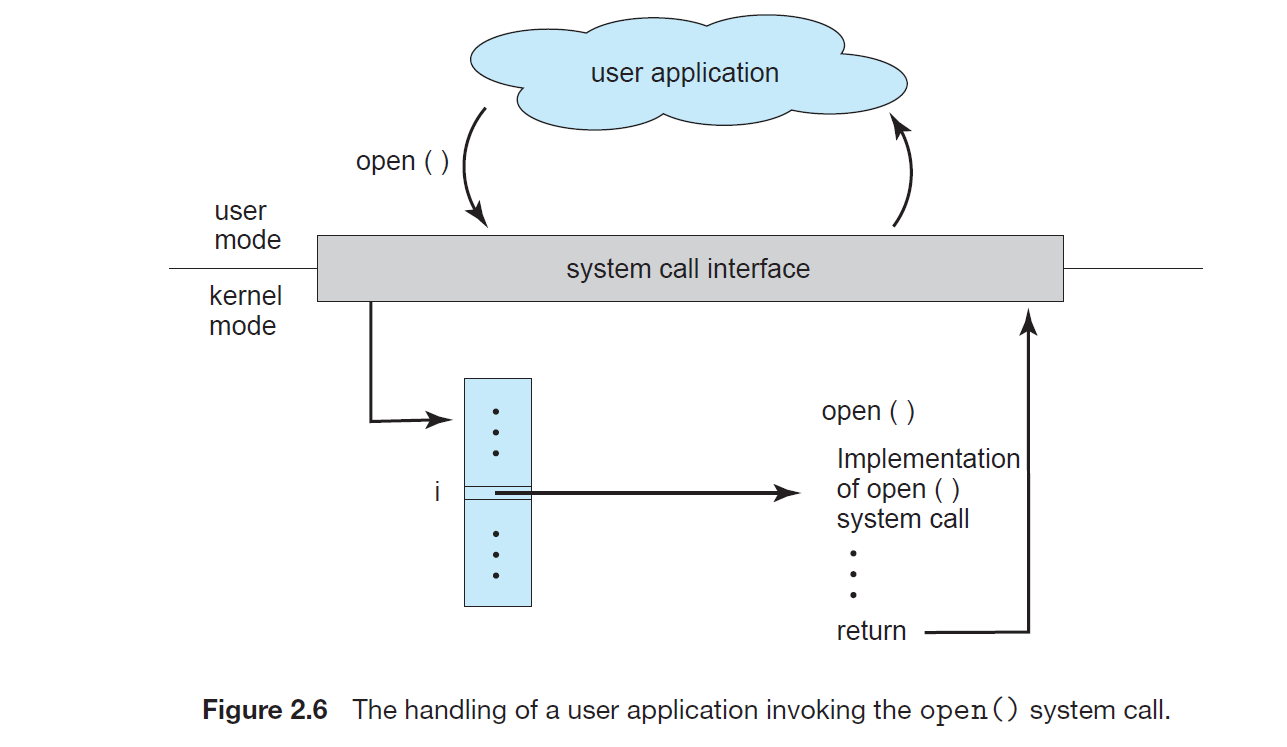
Read data from one file and copy them to another file-**copy file1.txt file2.txt**



**System Call Interface:**

For most programming languages provides the run-time support system. Run-time support system is a set of built-in library functions included with a compiler provides a system call interface. The System call interface that serves as the link to system calls made available by the operating system.

A Unique number is associated with each system call, and the system-call interface maintains a table indexed according to these numbers. The system call interface then invokes the intended system call in the operating-system kernel and returns the status of the system call and any return values.



**Pass Parameters to Operating System:**

The three general approaches for passing parameters to operating system are:

1.The simplest approach is to pass the parameters in registers.

2.In some cases, there may be more parameters than registers. In these cases, the parameters are generally stored in a block, or table, in memory, and the address of the block is passed as a parameter in a register.

3. Some operating systems prefer the block or stack method because those approaches do not limit the number or length of parameters being passed. Parameters can be placed or pushed onto the stack by the program and popped off the stack by the operating system.

