Practical 4 Required File Command

- **Open:** open and possibly create a file.
  Syntax: int open(const char *pathname, int flags,mode_t mode);

  The parameter flags is one of O_RDONLY, O_WRONLY or O_RDWR which request opening the file read-only, write-only or read/write respectively, bitwise-or'd with zero or more of the following.
  - O_CREATE:
    If the file does not exist it will be created. The owner (user id) of the file is set to the effective user id of the process.
  - O_TRUNC:
    If the file already exists and is a regular file and the open mode allows writing (i.e., is O_RDWR or O_WRONLY) it will be truncated to length 0.
  - O_APPEND:
    If the file is opened in append mode. Before each write, the file pointer is positioned at the end of the file.

- **Read Command:**
  Syntax: ssize_t read(int fd,void *buf, size_t count);

  Read() attempts to read up to count bytes from file descriptor fd into the buffer starting at buf.

- **Write Command:**
  Syntax: ssize_t write(int fd,void *buf, size_t count);

  write() writes up to count bytes from the buffer pointed buf to the file referred to by the file descriptor.

  fd: file pointer,
  count: writes up to count bytes to the file.

- **lseek command:**
  repositions the offset of the file descriptor fd to the argument offset.

  Syntax: off_t lseek(int fd,off_t offset,int whence);

  Whence
  SEEK_SET: the offset is set to offset bytes.
  SEEK_CUR: the offset is set to its current location plus offset bytes.
  SEEK_END: the offset is set to the size of the file plus offset bytes.

Program 1:

```
//This program opens a file, write to it and then read its contents
#include<stdio.h>
#include<fcntl.h>
#include<stdlib.h>
#include<string.h>

int main()
{
	char msg[81];
	int fd;

	fd=open("f1.txt",O_CREAT|O_WRONLY|O_TRUNC,0644);
	printf("Enter the message:");
	gets(msg);

	write(fd,msg,strlen(msg));
	close(fd);

	fd=open("f1.txt",O_RDONLY);
	read(fd,msg,msg);
	printf("Ur message:%s\n",msg);

	close(fd);
}
```

Program 2:

```
//This program copies a file to another
#include<stdio.h>
#include<fcntl.h>
int main()
{
	int fd1,fd2,n;
	char buf[80];

	fd1=open("f1.txt",O_RDONLY);
	fd2=open("f2.txt",O_CREAT|O_WRONLY|O_TRUNC,0644);

	while((n=read(fd1,buf,80))>0)
		write(fd2,buf,n);
	printf("Copy complete...\n");
}
```

```
//This program displays content of one file in reverse order into the another file

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

int main()
{
	int fd1,fd2;	//file descriptors
	char c;		//holds read char
	int offset;	//current offset

	fd1 = open("f1.txt", O_RDONLY);	//open file to read
	if ( fd1<0)
	{
		printf("%s", " Open Error");
	}

	fd2 = open("f2.txt", O_WRONLY | O_CREAT, 0670);	//open file to write
	if(fd2<0)
	{
		printf("%s", " Open Error");
	}

	offset = lseek(fd1,0,SEEK_END);	//go the end of file

	while (offset>0)
	{
		read(fd1, &c, 1);	//read a char
		write(fd2, &c, 1);	//write a char

		//go back to spot to the char before the one just read
		lseek(fd1, -2, SEEK_CUR);
		offset--;	//track the current offset
	}
	close(fd1);	//close the files
	close(fd2);

	return 0;
}
```