```c
/* Direct file I/O with fwrite() and fread(). */

#include <stdio.h>
#include <stdlib.h>

#define SIZE 20

main()
{
    int count, array1[SIZE], array2[SIZE];
    FILE *fp;

    /* Initialize array1[]. */

    for (count = 0; count < SIZE; count++)
        array1[count] = 2 * count;

    /* Open a binary mode file. */

    if ( (fp = fopen("direct.txt", "wb")) == NULL)
    {
        fprintf(stderr, "Error opening file.");
        exit(1);
    }
    /* Save array1[] to the file. */

    if (fwrite(array1, sizeof(int), SIZE, fp) != SIZE)
    {
        fprintf(stderr, "Error writing to file.");
        exit(1);
    }

    fclose(fp);

    /* Now open the same file for reading in binary mode. */

    if ( (fp = fopen("direct.txt", "rb")) == NULL)
    {
        fprintf(stderr, "Error opening file.");
        exit(1);
    }

    /* Read the data into array2[]. */

    if (fread(array2, sizeof(int), SIZE, fp) != SIZE)
    {
        fprintf(stderr, "Error reading file.");
        exit(1);
    }

    fclose(fp);

    /* Now display both arrays to show they're the same. */

    for (count = 0; count < SIZE; count++)
        printf("%d\t%d\n", array1[count], array2[count]);

}
```

```
/* Demonstrates ftell() and rewind(). */

#include <stdio.h>
#include <stdlib.h>
#define BUFLEN 6

char msg[] = "abcdefghijklmnopqrstuvwxyz";

main()
{
    FILE *fp;
    char buf[BUFLEN];

    if ( (fp = fopen("TEXT.TXT", "w")) == NULL)
    {
        fprintf(stderr, "Error opening file.");
        exit(1);
    }

    if (fputs(msg, fp) == EOF)
    {
        fprintf(stderr, "Error writing to file.");
        exit(1);
    }

    fclose(fp);

    /* Now open the file for reading. */

    if ( (fp = fopen("TEXT.TXT", "r")) == NULL)
    {
        fprintf(stderr, "Error opening file.");
        exit(1);
    }
    printf("\nImmediately after opening, position = %ld", ftell(fp));

    /* Read in 5 characters. */

    fgets(buf, BUFLEN, fp);
    printf("\nAfter reading in %s, position = %ld", buf, ftell(fp));

    /* Read in the next 5 characters. */

    fgets(buf, BUFLEN, fp);
    printf("\n\nThe next 5 characters are %s, and position now = %ld",
            buf, ftell(fp));

    /* Rewind the stream. */

    rewind(fp);

    printf("\n\nAfter rewinding, the position is back at %ld",
            ftell(fp));

    /* Read in 5 characters. */
    fgets(buf, BUFLEN, fp);
    printf("\nand reading starts at the beginning again: %s", buf);
    fclose(fp);
}
```

```
/* Detecting end-of-file. */

#include <stdio.h>
#include <stdlib.h>

#define BUFSIZE 100

main()
{
    char buf[BUFSIZE];
    char filename[20];
    FILE *fp;

    puts("Enter name of text file to display: ");
    gets(filename);

    /* Open the file for reading. */
    if ( (fp = fopen(filename, "r")) == NULL)
    {
        fprintf(stderr, "Error opening file.");
        exit(1);
    }

    /* If end of file not reached, read a line and display it. */

    while ( !feof(fp) )
    {
        fgets(buf, BUFSIZE, fp);
        printf("%s",buf);
    }

    fclose(fp);
}
```

```c
/* Random access with fseek(). */

#include <stdio.h>
#include <stdlib.h>
#include <io.h>

#define MAX 50

main()
{
    FILE *fp;
    int data, count, array[MAX];
    long offset;

    /* Initialize the array. */
    for (count = 0; count < MAX; count++)
        array[count] = count * 10;

    /* Open a binary file for writing. */
    if ( (fp = fopen("RANDOM.DAT", "wb")) == NULL)
    {
        fprintf(stderr, "\nError opening file.");
        exit(1);
    }

    /* Write the array to the file, then close it. */
    if ( (fwrite(array, sizeof(int), MAX, fp)) != MAX)
    {
        fprintf(stderr, "\nError writing data to file.");
        exit(1);
    }

    fclose(fp);

    /* Open the file for reading. */
    if ( (fp = fopen("RANDOM.DAT", "rb")) == NULL)
    {
        fprintf(stderr, "\nError opening file.");
        exit(1);
    }

    /* Ask user which element to read. Input the element */
    /* and display it, quitting when -1 is entered. */

    while (1)
    {
        printf("\nEnter element to read, 0-%d, -1 to quit: ",MAX-1);
        scanf("%ld", &offset);

        if (offset < 0)
            break;
        else if (offset > MAX-1)
            continue;

        /* Move the position indicator to the specified element. */
        if ( (fseek(fp, (offset*sizeof(int)), SEEK_SET)) != NULL)
        {
            fprintf(stderr, "\nError using fseek().");
            exit(1);
        }

        /* Read in a single integer. */
        fread(&data, sizeof(int), 1, fp);

        printf("\nElement %ld has value %d.", offset, data);
    }

    fclose(fp);
}
```