LAB-2: SAMPLE FILE-SYSTEM PROGRAMS

Description

The C library function **int fseek(FILE *stream, long int offset, int whence)** sets the file position of the **stream** to the given **offset**.

Declaration

Following is the declaration for fseek() function.

```
int fseek(FILE *stream, long int offset, int whence)
```

Parameters

- **stream** − This is the pointer to a FILE object that identifies the stream.
- **offset** − This is the number of bytes to offset from whence.
- **whence** − This is the position from where offset is added. It is specified by one of the following constants −

SL.No.	Constant & Description
1	**SEEK_SET** ~ Beginning of file
2	**SEEK_CUR** ~ Current position of the file pointer
3	**SEEK_END** ~ End of file

Return Value

This function returns zero if successful, or else it returns a non-zero value.

Example

The following example shows the usage of fseek() function.

```
#include <stdio.h>

int main ()
{
   FILE *fp;

   fp = fopen("file.txt","w+");
   fputs("This is tutorialspoint.com", fp);

   fseek( fp, 7, SEEK_SET );
   fputs(" C Programming Language", fp);
```

```
  fclose(fp);
  return(0);
}
```

Let us compile and run the above program that will create a file **file.txt** with the following content. Initially program creates the file and writes *This is tutorialspoint.com* but later we had reset the write pointer at 7th position from the beginning and used puts() statement which over-write the file with the following content −

```
This is C Programming Language
```

Now let's see the content of the above file using the following program −

```
#include <stdio.h>

int main () {
  FILE *fp;
  int c;

  fp = fopen("file.txt","r");
  while(1) {
    c = fgetc(fp);
    if( feof(fp) ) {
      break;
    }
    printf("%c", c);
  }
  fclose(fp);
  return(0);
}
```

Let us compile and run the above program to produce the following result −

```
This is C Programming Language
```

fseek() in C/C++ with example

fseek() is used to move file pointer associated with a given file to a specific position.

Syntax:

int fseek(FILE *pointer, long int offset, int position)

- **pointer:** pointer to a FILE object that identifies the stream.
- **offset:** number of bytes to offset from position
- **position:** position from where offset is added.

Returns:

zero if successful, or else it returns a non-zero value

position defines the point with respect to which the file pointer needs to be moved. It has three values:

- **SEEK_END :** It denotes end of the file.
- **SEEK_SET :** It denotes starting of the file.
- **SEEK_CUR :** It denotes file pointer's current position.

```
// C Program to demonstrate the use of fseek()

#include <stdio.h>

int main() {
   FILE *fp;
   fp = fopen("test.txt", "r");

   // Moving pointer to end
   fseek(fp, 0, SEEK_END);

   // Printing position of pointer
   printf("%ld", ftell(fp));

   return 0;
}
```

Output:
81

LAB-2: SAMPLE FILE-SYSTEM PROGRAMS

1. C program to read name and marks of n number of students and store them in a file.

```
#include <stdio.h>
int main()
{
  char name[50];
  int marks, i, num;

  printf("Enter number of students: ");
  scanf("%d", &num);

  FILE *fptr;
  fptr = (fopen("C:\\student.txt", "w"));
  if(fptr == NULL)
  {
     printf("Error!");
     exit(1);
  }

  for(i = 0; i < num; ++i)
  {
    printf("For student%d\nEnter name: ", i+1);
    scanf("%s", name);

    printf("Enter marks: ");
    scanf("%d", &marks);

    fprintf(fptr,"\nName: %s \nMarks=%d \n", name, marks);
  }

  fclose(fptr);
```

2. C program to read name and marks of n number of students from and store them in a file. If the file previously exits, add the information to the file.

```
#include <stdio.h>
int main()
{
  char name[50];
```

```
  int marks, i, num;

  printf("Enter number of students: ");
  scanf("%d", &num);

  FILE *fptr;
  fptr = (fopen("C:\\student.txt", "a"));
  if(fptr == NULL)
  {
      printf("Error!");
      exit(1);
  }

  for(i = 0; i < num; ++i)
  {
    printf("For student%d\nEnter name: ", i+1);
    scanf("%s", name);

    printf("Enter marks: ");
    scanf("%d", &marks);

    fprintf(fptr,"\nName: %s \nMarks=%d \n", name, marks);
  }

  fclose(fptr);
  return 0;
}
```

3. C program to write all the members of an array of structures to a file using fwrite(). Read the array from the file and display on the screen.

```
#include <stdio.h>
struct student
{
  char name[50];
  int height;
};
int main(){
   struct student stud1[5], stud2[5];
   FILE *fptr;
```

```
    int i;

    fptr = fopen("file.txt","wb");
    for(i = 0; i < 5; ++i)
    {
        fflush(stdin);
        printf("Enter name: ");
        gets(stud1[i].name);

        printf("Enter height: ");
        scanf("%d", &stud1[i].height);
    }

    fwrite(stud1, sizeof(stud1), 1, fptr);
    fclose(fptr);

    fptr = fopen("file.txt", "rb");
    fread(stud2, sizeof(stud2), 1, fptr);
    for(i = 0; i < 5; ++i)
    {
        printf("Name: %s\nHeight: %d", stud2[i].name, stud2[i].height);
    }
    fclose(fptr);
}
```