

Advanced Unix Programming
Assignment 1
Vijesh Ghandare - 111403013

1. Assume that you have to read 10 characters from the beginning of an existing file and then to write “hello” to the end of the file. Write a program to achieve this without using lseek function.

Program:

```
#include<stdio.h>
#include<sys/types.h>
#include<stdlib.h>
#include<fcntl.h>
#include<unistd.h>
#include<errno.h>
#include <sys/stat.h>
int main(){
    int fd = open("file",O_RDWR | O_APPEND);
    if(fd == -1){
        perror("open failed..!!!");
        return errno;
    }
    char a[10];
    int r = read(fd,a,10);
    if(r <= 0){
        printf("read failed..");
    }
    int w = write(fd,"hello",5);
    if(w != 5){
        printf("write failed..");
    }
    return 0;
}
```

Output:

```
> cat file
This is the content of file.

> cc Lab1.1.c
> ./a.out
> cat file
This is the content of file.hello
```

**2. Linux provides a function as given below to truncate file to specific length.
int truncate (const char *path, off_t len); return 0 on success. On error, return -1,
Write a program to emulate this function. Use cat command to demonstrate.**

Program:

```
#include <stdio.h>
#define _XOPEN_SOURCE_EXTENDED 1
#include <unistd.h> // for truncate
#include <stdlib.h> // for system call
#include <sys/types.h>
#include <fcntl.h>
#include <sys/stat.h>

int truncat(char *path_name, int size) {
    int cnt;
    struct stat st;
    int file_size;
    char n = 0;
    char command[50];
    int fd;
    char *buffer;

    snprintf(command, sizeof(command), "od -c %s", path_name);

    if(access(path_name, W_OK) == 0) {
        stat(path_name, &st);
        file_size = st.st_size;
        fd = open(path_name, O_RDWR);
        if(fd == 0) {
            printf("Error opening file\n");
            return -1;
        }
        if(file_size >= size){
            buffer = (char *)malloc(size+1);
            read(fd, buffer, size + 1);
            close(fd);
            fd = open(path_name, O_WRONLY | O_TRUNC);
            write(fd, buffer, size);
            printf("#File name :%s \n#File contents after truncate: \n ", path_name);
            system(command);
            printf("\n");
            close(fd);
        }
        else{
            cnt = 0;
            int temp = (size - file_size);
            fd = open(path_name, O_WRONLY);
            lseek(fd, 0, SEEK_END);
            //write(fd, buffer, file_size);
            while(cnt < temp) {
                write(fd,&n,1);
            }
        }
    }
}
```

```

        cnt++;
    }
    printf("#File name :%s \n#File contents after truncate: \n ", path_name);
    system(command);
    printf("\n");
    close(fd);
}
} else {
    printf("File dont have write permission");
    return -1;
}
return 0;
}

int main() {
    char path_name[50];
    char command[50];
    int size;
    printf("#Please enter path :");
    scanf("%s", path_name);
    printf("#File name : %s \n#File contents before truncate: \n ", path_name);
    snprintf(command, sizeof(command), "od -c %s", path_name);
    system(command);
    struct stat st;
    stat(path_name, &st);
    printf("Original size = %zu\n", st.st_size);
    printf("#Please enter number to truncate file :");
    scanf("%d",&size);
    truncat(path_name, size);
    return 0;
}

```

Output:

Case1: When entered size is more than original file size.

(Adding “null characters”)

```

vijesh1996@vijesh1996-HP-Pavilion-15-Notebook-PC:~/Desktop/AUP_LAB/Lab1$ ./a.out
#Please enter path :file
#File name : file
#File contents before truncate:
00000000  T  h  i  s      i  s      t  h  e      c  o  n  t
00000020  e  n  t      o  f      f  i  l  e      b  e  f  o
00000040  r  e      e  x  e  c  u  t  i  o  n      o  f
00000060  p  r  o  g  r  a  m  .  \n
00000071
Original size = 57
#Please enter number to truncate file :70
#File name :file
#File contents after truncate:
00000000  T  h  i  s      i  s      t  h  e      c  o  n  t
00000020  e  n  t      o  f      f  i  l  e      b  e  f  o
00000040  r  e      e  x  e  c  u  t  i  o  n      o  f
00000060  p  r  o  g  r  a  m  .  \n  \0  \0  \0  \0  \0  \0  \0
0000100  \0  \0  \0  \0  \0  \0
0000106

```

Case 2: When entered size is less than original file size.

```
vtjesh1996@vtjesh1996-HP-Pavilion-15-Notebook-PC:~/Desktop/AUP_LAB/Lab1$ ./a.out
#Please enter path :file
#File name : file
#File contents before truncate:
00000000 T h i s i s t h e c o n t
00000020 e n t o f f i l e b e f o
00000040 r e e x e c u t i o n o f
00000060 p r o g r a m . \n \0 \0 \0 \0 \0 \0
0000100 \0 \0 \0 \0 \0 \0
0000106
Original size = 70
#Please enter number to truncate file :57
#File name :file
#File contents after truncate:
00000000 T h i s i s t h e c o n t
00000020 e n t o f f i l e b e f o
00000040 r e e x e c u t i o n o f
00000060 p r o g r a m . \n
00000071
```

3. What will be the output for the program with following operation?

- Create a new file "f1" and write "abcde" in it and close
- Open the file "f1" for writing with O_APPEND flag
- lseek to the beginning of the file
- Replace the existing data in the file with "12345"

Justify your answer.

Program:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<unistd.h>
#include<errno.h>
int main() {
    int fd;
    fd = open("f1",O_WRONLY | O_CREAT,S_IRUSR | S_IWUSR);
    if(fd == -1){
        perror("error opening file.");
        return errno;
    }
    if(write(fd,"abcde",5) != 5){
        perror("error while writing..");
        return errno;
    }
    close(fd);
    fd = open("f1", O_WRONLY | O_APPEND);
    if(lseek(fd,0,SEEK_SET) != 0)
        printf("LSEEK failed...!!\n");
    if(write(fd,"12345",5) != 5){
        perror("error writing to file");
        return errno;
    }
    close(fd);
    return 0;
}
```

Output:

```
> ./a.out
> cat f1
abcde12345
```

4. Write a program to create a file with a hole: write any 10 bytes at an offset of 10 and another 10 bytes at an offset of 30. Using “system” function, invoke “od” command and view the contents. Later copy the contents of the file to another file without writing the bytes of 0. Once again verify the contents by invoking “system” with “od”.

Program:

```
/*> ./a.out
> ls -l file1.txt
> ls -l file2.txt
> od -c file1.txt
> od -c file2.txt
*/
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<unistd.h>
#include<stdlib.h>
int main(void){
    int fd2;
    char buff3[2];
    char buf1[] = "abcdefghij";
    char buf2[] = "ABCDEFGHIIJ";
    int fd1 = open("file1.txt",O_RDWR);
    if ( fd1 < 0)
        printf("open failed..!!!");
    if (write(fd1, buf1, 10) != 10)
        printf("buf1 write error");

    if (lseek(fd1, 30, SEEK_SET) == -1)
        printf("lseek failed..!!!");

    if (write(fd1, buf2, 10) != 10)
        printf("buf2 write failed..!!!");

    fd2 = open("file2.txt", O_WRONLY);
    if(fd2 < 0)
        printf("open failed..!!!");

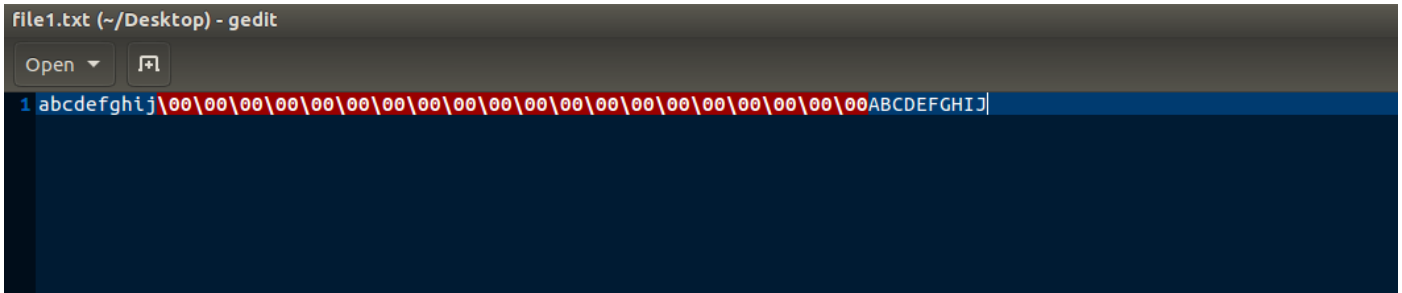
    if(lseek(fd1,0,SEEK_SET) != 0)
        printf("LSEEK failed..\n");
    while(read(fd1,buff3,1) > 0){
        if(buff3[0] != 0)
            write(fd2,buff3,1);
    }
}
```

```
close(fd1);
close(fd2);
```

```
}
```

Output:

I) Content of File 1:



```
file1.txt (~/Desktop) - gedit
Open  [icon]
1 abcdefghij\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0ABCDEFGHIJ
```

II) Content of File 2: Initially File 2 is empty.

III) Copying the content of File 1 to File 2 without holes:

```
viresh1996@viresh1996-HP-Pavilion-15-Notebook-PC:~/Desktop$ cc Lab14.c
viresh1996@viresh1996-HP-Pavilion-15-Notebook-PC:~/Desktop$ ./a.out
viresh1996@viresh1996-HP-Pavilion-15-Notebook-PC:~/Desktop$ od -c file1.txt
00000000  a  b  c  d  e  f  g  h  i  j  \0 \0 \0 \0 \0 \0
00000020  \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0  A  B
00000040  C  D  E  F  G  H  I  J
00000050
viresh1996@viresh1996-HP-Pavilion-15-Notebook-PC:~/Desktop$ od -c file2.txt
00000000  a  b  c  d  e  f  g  h  i  j  A  B  C  D  E  F
00000020  G  H  I  J
00000024
viresh1996@viresh1996-HP-Pavilion-15-Notebook-PC:~/Desktop$
```

IV) Content of File 2 after copying the content of File 1 without holes:



```
file2.txt (~/Desktop) - gedit
Open  [icon]
1 abcdefghijABCDEFGHIJ
```