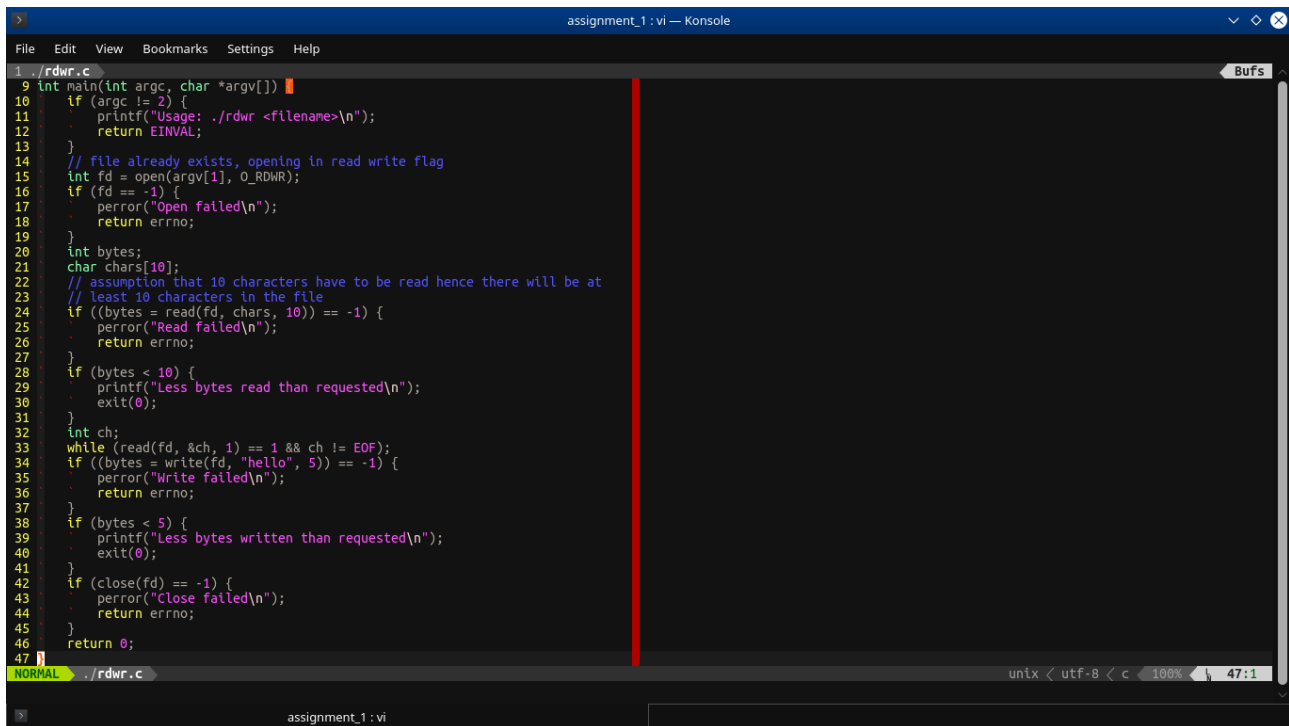


# Assignment 1

Srishti Shelke 111603056  
Niramay Vaidya 111605075

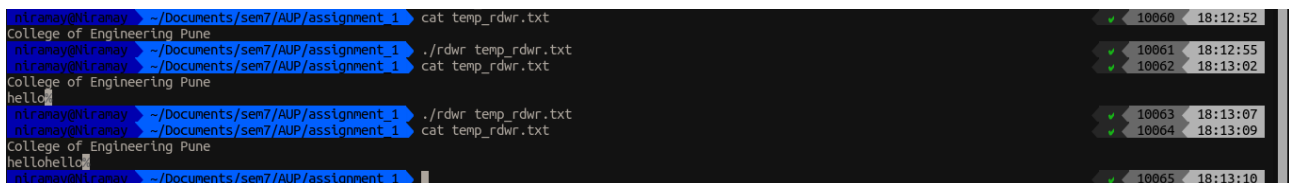
1. Assume that you have to read 10 characters from the beginning of an existing file and then to write "hello" to the end of the file. Write a program to achieve this without using lseek function.

Code-



```
1 ./rdwr.c
9 int main(int argc, char *argv[])
10 {
11     if (argc != 2) {
12         printf("Usage: ./rdwr <filename>\n");
13         return EINVAL;
14     }
15     // file already exists, opening in read write flag
16     int fd = open(argv[1], O_RDWR);
17     if (fd == -1) {
18         perror("Open failed\n");
19         return errno;
20     }
21     int bytes;
22     char chars[10];
23     // assumption that 10 characters have to be read hence there will be at
24     // least 10 characters in the file
25     if ((bytes = read(fd, chars, 10)) == -1) {
26         perror("Read failed\n");
27         return errno;
28     }
29     if (bytes < 10) {
30         printf("Less bytes read than requested\n");
31         exit(0);
32     }
33     int ch;
34     while (read(fd, &ch, 1) == 1 && ch != EOF);
35     if ((bytes = write(fd, "hello", 5)) == -1) {
36         perror("Write failed\n");
37         return errno;
38     }
39     if (bytes < 5) {
40         printf("Less bytes written than requested\n");
41         exit(0);
42     }
43     if (close(fd) == -1) {
44         perror("Close failed\n");
45         return errno;
46     }
47     return 0;
48 }
```

Output-



```
niranay@niranay: ~/Documents/sem7/AUP/assignment_1
niranay@niranay:~/Documents/sem7/AUP/assignment_1$ cat temp_rdwr.txt
College of Engineering Pune
niranay@niranay:~/Documents/sem7/AUP/assignment_1$ ./rdwr temp_rdwr.txt
niranay@niranay:~/Documents/sem7/AUP/assignment_1$ cat temp_rdwr.txt
hellohello
niranay@niranay:~/Documents/sem7/AUP/assignment_1$
```

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 (donot use any built-in function). Use cat command to demonstrate the contents of the truncated file.

## Code-

```
assignment_1:vi — Konsole
File Edit View Bookmarks Settings Help
1 ./truncate.c
9 int mytruncate(int fd, int offset)
10 {
11     int size = lseek(fd, 0, SEEK_END);
12     if (size == -1) {
13         perror("lseek failed in truncate\n");
14         return 1;
15     }
16     if (size == offset) {
17         return 0;
18     }
19     else if (size < offset) {
20         if (lseek(fd, 1, SEEK_END) == -1) {
21             return 1;
22         }
23         char ch = '\0';
24         for (int i = offset - size; i > 0; i--) {
25             if (write(fd, &ch, 1) != 1) {
26                 return 1;
27             }
28         }
29         return 0;
30     }
31     else { // size > offset
32         int pos;
33         if ((pos = lseek(fd, offset, SEEK_SET)) == -1) {
34             return 1;
35         }
36         char ch = '\0';
37         for (int i = pos; i <= size; i++) {
38             if (write(fd, &ch, 1) != 1) {
39                 return 1;
40             }
41         }
42         return 0;
43     }
44 }
NORMAL ./truncate.c
unix < utf-8 < c 67% 44:1
```

```
assignment_1:vi — Konsole
File Edit View Bookmarks Settings Help
1 ./truncate.c
45
46 int main(int argc, char *argv[]) {
47     if (argc != 3) {
48         printf("Usage: ./truncate <filename> <offset>\n");
49         return EINVAL;
50     }
51     // file already exists, opening in read write flag
52     int fd = open(argv[1], O_RDWR);
53     if (fd == -1) {
54         perror("Open failed\n");
55         return errno;
56     }
57     if (mytruncate(fd, atoi(argv[2])) == 1) {
58         printf("Truncate failed\n");
59         exit(0);
60     }
61     if (close(fd) == -1) {
62         perror("Close failed\n");
63         return errno;
64     }
65     return 0;
66 }
```

## Output-

```
niranay@niranay: ~/Documents/sem7/AUP/assignment_1
College of Engineering Pune
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ cat temp_truncate.txt
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ ./truncate temp_truncate.txt 20
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ cat temp_truncate.txt
College of Engineering Pune
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ od -c temp_truncate.txt
00000000  C o l l e g e o f E n g i n
00000020  e e r i \0 \0 \0 \0 \0 \0 \0 \0
00000035
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ ./truncate temp_truncate.txt 50
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ cat temp_truncate.txt
College of Engineering Pune
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$ od -c temp_truncate.txt
00000000  C o l l e g e o f E n g i n
00000020  e e r i \0 \0 \0 \0 \0 \0 \0 \0
00000040  \0 \0 \0 \0 \0 \0 \0 \0
00000060  \0 \0 \0
00000064
niranay@niranay: ~/Documents/sem7/AUP/assignment_1$
```

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.

Code-

```
1 ./append.c
9 int main(int argc, char *argv[]) {
10     if (argc != 2) {
11         printf("Usage: ./append <filename>\n");
12         return EINVAL;
13     }
14     int fd = open(argv[1], O_WRONLY | O_CREAT, S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH);
15     if (fd == -1) {
16         perror("Open failed\n");
17         return errno;
18     }
19     int bytes;
20     if ((bytes = write(fd, "abcde", 5)) == -1) {
21         perror("Write failed\n");
22         return errno;
23     }
24     if (bytes < 5) {
25         printf("Less bytes written than requested\n");
26         exit(0);
27     }
28     if (close(fd) == -1) {
29         perror("Close failed\n");
30         return errno;
31     }
32     int fd_new = open(argv[1], O_WRONLY | O_APPEND, S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH);
33     if (fd_new == -1) {
34         perror("Open failed\n");
35         return errno;
36     }
37     if (lseek(fd, 0, SEEK_SET) == -1) {
38         perror("Lseek failed\n");
39         return errno;
40     }
41     if ((bytes = write(fd_new, "12345", 5)) == -1) {
42         perror("Write failed\n");
43         return errno;
44     }
45 }
NORMAL ./append.c
```

```
1 ./append.c
45     if (bytes < 5) {
46         printf("Less bytes written than requested\n");
47         exit(0);
48     }
49     if (close(fd_new) == -1) {
50         perror("Close failed\n");
51         return errno;
52     }
53     return 0;
54 }
```

Output-

```
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$ ./append temp_append.txt
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$ ll temp_append.txt
-rw-rw-r-- 1 niranmay niranmay 10 Oct 20 18:27 temp_append.txt
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$ cat temp_append.txt
abcde12345
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$ ./append temp_append.txt
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$ cat temp_append.txt
abcde1234512345
niranmay@niranmay: ~/Documents/sem7/AUP/assignment_1$
```

Explanation-

When the file is opened with the O\_APPEND mode, calling lseek to change the file offset does not have any effect in terms of writing to the lseeked position. Irrespective of the successful call to lseek, when such a file is written into, the data always gets appended to the end of the file and does not replace existing data at the lseeked position.

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”.

## Code-

```
assignment_1: vi — Konsole
File Edit View Bookmarks Settings Help
1 ./hole_in_file.c
9 int main(int argc, char *argv[]) {
10     if (argc != 3) {
11         printf("Usage: ./hole_in_file <filename with hole> <filename to copy to without hole>\n");
12     }
13     int fd = open(argv[1], O_RDWR | O_CREAT, S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH);
14     if (fd == -1) {
15         perror("Open failed\n");
16         return errno;
17     }
18     if (lseek(fd, 10, SEEK_SET) == -1) {
19         perror("Lseek failed\n");
20         return errno;
21     }
22     int bytes;
23     if ((bytes = write(fd, "#####", 10)) == -1) {
24         perror("Write failed\n");
25         return errno;
26     }
27     if (bytes < 10) {
28         printf("Less bytes written than requested\n");
29         exit(0);
30     }
31     if (lseek(fd, 30, SEEK_SET) == -1) {
32         perror("Lseek failed\n");
33         return errno;
34     }
35     if ((bytes = write(fd, "#####", 10)) == -1) {
36         perror("Write failed\n");
37         return errno;
38     }
39     if (bytes < 10) {
40         printf("Less bytes written than requested\n");
41         exit(0);
42     }
43     if (lseek(fd, 0, SEEK_SET) == -1) {
44         perror("Lseek failed\n");
45     }
46 }
NORMAL ./hole_in_file.c
unix < utf-8 < c 35% 27:21
```

```
assignment_1: vi — Konsole
File Edit View Bookmarks Settings Help
1 ./hole_in_file.c
45 }
46 system("od -c temp_file_with_hole.txt");
47 int fd_new = open(argv[2], O_WRONLY | O_CREAT, S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH);
48 if (fd_new == -1) {
49     perror("Open failed\n");
50     return errno;
51 }
52 char ch;
53 while (read(fd, &ch, 1) == 1 && ch != EOF) {
54     if (ch != '\0') {
55         if (bytes = write(fd_new, &ch, 1) == -1) {
56             perror("Write failed\n");
57             return errno;
58         }
59     }
60     if (bytes < 1) {
61         printf("Less bytes written than requested\n");
62         exit(0);
63     }
64 }
65 }
66 system("od -c temp_file_without_hole.txt");
67 if (close(fd) == -1) {
68     perror("Close failed\n");
69     return errno;
70 }
71 if (close(fd_new) == -1) {
72     perror("Close failed\n");
73     return errno;
74 }
75 return 0;
76 }
77 }
NORMAL ./hole_in_file.c
unix < utf-8 < c 81% 62:24
```

## Output-

```
root@kali:~/Documents/sem7/AUP/assignment_1# ./hole_in_file temp_file_with_hole.txt temp_file_without_hole.txt
00000000 \0 \0 \0 \0 \0 \0 \0 \0 \0 # # # # #
00000020 # # # # \0 \0 \0 \0 \0 \0 \0 \0 \0 # #
00000040 # # # # # # # #
00000060
00000080 # # # # # # # # # # # # # # #
000000A0 # # # #
000000C0
000000E0
```

root@kali:~/Documents/sem7/AUP/assignment\_1#