190905104
Parth Shukla
Lab 5

1) Write a producer and consumer program in C using the FIFO queue. The producer should write a set of 4 integers into the FIFO queue and the consumer should display the 4 integers.

```
// Producer

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

#define FIFO_NAME "my_fifo"
#define BUFFER_SIZE 1000


int main(int argc, char *argv[]){
    int pipe_fd;
    int res;
    int open_mode=O_WRONLY;
    int n=0;
    char buffer[BUFFER_SIZE+1];

    if(access(FIFO_NAME,F_OK)==-1){
        res=mkfifo(FIFO_NAME,0777);
        if(res!=0){
            fprintf(stderr, "Could not create file%s\n",FIFO_NAME );
            exit(EXIT_FAILURE);
        }
    }

    printf("Process %d opening FIFO O_WRONLY\n",getpid());
    pipe_fd = open(FIFO_NAME,open_mode);
    printf("Process %d result %d\n",getpid(),pipe_fd);

    if (pipe_fd!=-1){
        printf("Enter 4 numbers\n");

        while(n<4){
            scanf("%s",buffer);
            res=write(pipe_fd,buffer,BUFFER_SIZE);

            if(res==-1){
                fprintf(stderr, "Write Error on Pipe\n");
                exit(EXIT_FAILURE);
            }
            n++;
        }
        (void)close(pipe_fd);
    }
```

```
    else
        exit(EXIT_FAILURE);

    printf("Process %d Finished\n",getpid() );
    exit(EXIT_SUCCESS);

}

// Consumer

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

#define FIFO_NAME "my_fifo"
#define BUFFER_SIZE 1000

int main(int argc, char *argv[]){
    int pipe_fd;
    int res;
    int open_mode=O_RDONLY;
    int n=0;
    char buffer[BUFFER_SIZE+1];
    memset(buffer,'\0',sizeof(buffer));

    printf("Process %d opening FIFO O_RDONLY\n",getpid());
    pipe_fd=open(FIFO_NAME,open_mode);
    printf("Process %d result %d\n",getpid(),pipe_fd);

    if (pipe_fd!=-1){
        do{
            res=read(pipe_fd,buffer,BUFFER_SIZE);
            printf("%s\n",buffer );
            n++;
        }while(n<4);

        (void)close(pipe_fd);
    }
    else
        exit(EXIT_FAILURE);

    printf("Process %d Finished, %d bytes read\n",getpid(),n );
    exit(EXIT_SUCCESS);
}
```

```
Student@project-lab:~/190905104_OS/lab5$ gcc prod.c -o producer
Student@project-lab:~/190905104_OS/lab5$ ./producer
Process 5338 opening FIFO O_WRONLY
Process 5338 result 3
Enter 4 numbers
3
7
2
9
Process 5338 Finished
Student@project-lab:~/190905104_OS/lab5$
```

```
Student@project-lab:~/190905104_OS/lab5$ gcc cons.c -o consumer
Student@project-lab:~/190905104_OS/lab5$ ./consumer
Process 5393 opening FIFO O_RDONLY
Process 5393 result 3
3
7
2
9
Process 5393 Finished, 4 bytes read
Student@project-lab:~/190905104_OS/lab5$
```

2) Demonstrate creation, writing to, and reading from a pipe.

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
#include<limits.h>
#include<fcntl.h>
#include<sys/msg.h>
#include<sys/stat.h>
#include<string.h>

int main(int argc, char* argv[]){
    int n;
    int fd[2];
    char buf[1024];
    char *data = "Hello from q2.c";
    pipe(fd);

    write(fd[1], data, strlen(data));
    if((n=read(fd[0], buf, 1023)) >= 0){
        buf[n] = '\0';
        printf("Read %d bytes\n%s\n", n, buf);
    }else{
        perror("Read");
        exit(0);
    }
}
```

```
Student@project-lab:~/190905104_OS/lab5$ ./q2
Read 15 bytes
Hello from q2.c
```

3) Write a C program to implement one side of FIFO.

```
// User 1 – First writes and then reads

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

#define FIFO_NAME "my_fifo"
#define BUFFER_SIZE 1000
```

```
int main(int argc, char *argv[]){
    int pipe_fd;
    int res;
    int open_mode_write = O_WRONLY;
    int open_mode_read = O_RDONLY;
    int n=0;
    char buffer[BUFFER_SIZE+1];

    if(access(FIFO_NAME,F_OK)==-1){
        res=mkfifo(FIFO_NAME,0777);
        if(res!=0){
            fprintf(stderr, "Could not create file%s\n",FIFO_NAME );
            exit(EXIT_FAILURE);
        }
    }



    while(1){
        printf("Current mode: Writing\n");
        pipe_fd = open(FIFO_NAME, open_mode_write);
        printf("Enter message: ");
        fgets(buffer, BUFFER_SIZE, stdin);
        write(pipe_fd, buffer, BUFFER_SIZE);
        close(pipe_fd);
        printf("\n");


        printf("Current mode: Reading\n");
        pipe_fd = open(FIFO_NAME, open_mode_read);
        res = read(pipe_fd, buffer, BUFFER_SIZE);
        if(res == -1){
            perror("Read");
            exit(0);
        }
        printf("%s\n", buffer);
        close(pipe_fd);

    }
    (void)close(pipe_fd);

    printf("Process %d Finished\n",getpid() );
    exit(EXIT_SUCCESS);
}

// User 2 – First reads and then writes back

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

#define FIFO_NAME "my_fifo"
#define BUFFER_SIZE 1000
```

```
int main(int argc, char *argv[]){
    int pipe_fd;
    int res;
    int open_mode_write = O_WRONLY;
    int open_mode_read = O_RDONLY;
    int n=0;
    char buffer[BUFFER_SIZE+1];

    if(access(FIFO_NAME,F_OK)==-1){
        res=mkfifo(FIFO_NAME,0777);
        if(res!=0){
            fprintf(stderr, "Could not create file%s\n",FIFO_NAME );
            exit(EXIT_FAILURE);
        }
    }



    while(1){

        printf("Current mode: Reading\n");
        pipe_fd = open(FIFO_NAME, open_mode_read);
        res = read(pipe_fd, buffer, BUFFER_SIZE);
        if(res == -1){
            perror("Read");
            exit(0);
        }
        printf("%s\n", buffer);
        close(pipe_fd);


        printf("Current mode: Writing\n");
        pipe_fd = open(FIFO_NAME, open_mode_write);
        printf("Enter message: ");
        fgets(buffer, BUFFER_SIZE, stdin);
        write(pipe_fd, buffer, BUFFER_SIZE);
        close(pipe_fd);
        printf("\n");
    }
    (void)close(pipe_fd);

    printf("Process %d Finished\n",getpid() );
    exit(EXIT_SUCCESS);
}
```

```
Student@project-lab:~/190905104_OS/lab5$ gcc q3_1.c -o u1
Student@project-lab:~/190905104_OS/lab5$ ./u1
Current mode: Writing
Enter message: hello from first

Current mode: Reading
hello from second

Current mode: Writing
Enter message: lab

Current mode: Reading
5

Current mode: Writing
Enter message: done

Current mode: Reading
```

```
Student@project-lab:~/190905104_OS/lab5$ gcc q3_2.c -o u2
Student@project-lab:~/190905104_OS/lab5$ ./u2
Current mode: Reading
hello from first

Current mode: Writing
Enter message: hello from second

Current mode: Reading
lab

Current mode: Writing
Enter message: 5

Current mode: Reading
done

Current mode: Writing
Enter message:
```

4) Write a C program reading and writing a binary file in C.

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

int main(){
    FILE* fptr;
    int num=0;
    fptr=fopen("q4.bin","wb+");

    printf("Enter 5 numbers: \n");

    for(int i = 0; i < 5; i++){
        scanf("%d",&num);
        fwrite(&num, sizeof(int), 1, fptr);
    }

    printf("Writing done!\n");
    fclose(fptr);

    fptr=fopen("q4.bin","rb");

    for(int i = 0; i < 5; i++){
        fread(&num,sizeof(int),1,fptr);
        printf("%d\n",num);
    }

}
```

```
Student@project-lab:~/190905104_OS/lab5$ ./q4
Enter 5 numbers:
1
2
3
4
5
Writing done!
1
2
3
4
5
Student@project-lab:~/190905104_OS/lab5$
```