

Q1. 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.c
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <limits.h>
#include <sys/types.h>
#include <sys/stat.h>

#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF
#define TEN_MEG (1024 * 1024 * 10)

int main()
{
    int pipe_fd;
    int res;
    int open_mode = O_WRONLY;
    int bytes_sent = 0;
    int buffer[BUFFER_SIZE + 1];
    if (access(FIFO_NAME, F_OK) == -1)
    {
        res = mkfifo(FIFO_NAME, 0777);
        if (res != 0)
        {
            fprintf(stderr, "Could not create fifo %s\n", FIFO_NAME);
            exit(EXIT_FAILURE);
        }
    }
    printf("Process %d opening FIFO O_WRONLY\n", getpid());
    printf(" Enter 4 integers:\t");
    for (int i = 0; i < 4; i++)
        scanf("%d", &buffer[i]);
    pipe_fd = open(FIFO_NAME, open_mode);
    printf("Process %d result %d\n", getpid(), pipe_fd);
    if (pipe_fd != -1)
    {
        while(bytes_sent < TEN_MEG)
        {
            res = write(pipe_fd, buffer, BUFFER_SIZE);
            if (res == -1)
            {
                fprintf(stderr, "Write error on pipe\n");
                exit(EXIT_FAILURE);
            }
            bytes_sent += res;
        }
        (void)close(pipe_fd);
    }
    else
        exit(EXIT_FAILURE);
    printf("Process %d finished\n", getpid());
    exit(EXIT_SUCCESS);
}
```

```

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

#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF

int main()
{
    int pipe_fd;
    int res;
    int open_mode = O_RDONLY;
    int buffer[BUFFER_SIZE + 1];
    int bytes_read = 0;
    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);
            bytes_read += res;
        } while (res > 0);
        (void)close(pipe_fd);
        for (int i = 0; i < 4; i++)
            printf("%d ", buffer[i]);
        printf("\n");
    }
    else
        exit(EXIT_FAILURE);
    printf("Process %d finished, %d bytes read\n", getpid(), bytes_read);
    exit(EXIT_SUCCESS);
}

```

```

180905380@prg08: ~/Desktop/Operating Systems/Week 5
File Edit View Search Terminal Help
180905380@prg08:~/Desktop/Operating Systems/Week 5$ ./o1p
Process 12381 opening FIFO O_RDONLY
Enter 4 integers: 1 2 3 -5
Process 12381 result 3
Process 12381 finished
180905380@prg08:~/Desktop/Operating Systems/Week 5$

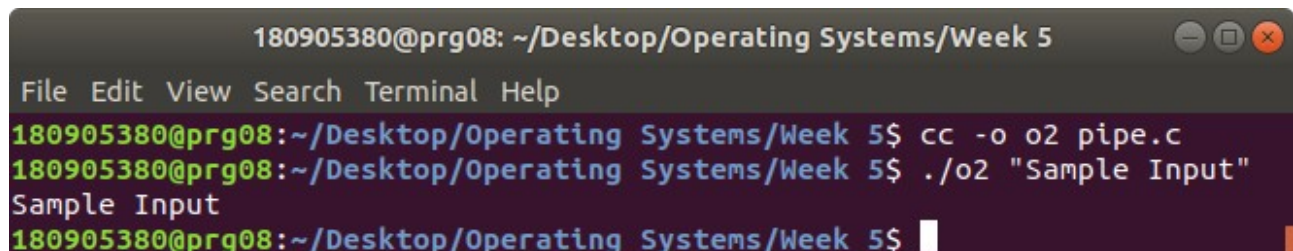
180905380@prg08: ~/Desktop/Operating Systems/Week 5
File Edit View Search Terminal Help
180905380@prg08:~/Desktop/Operating Systems/Week 5$ cc -o o1c consumer.c
180905380@prg08:~/Desktop/Operating Systems/Week 5$ ./o1c
Process 12382 opening FIFO O_RDONLY
Process 12382 result 3
1 2 3 -5
Process 12382 finished, 10485760 bytes read
180905380@prg08:~/Desktop/Operating Systems/Week 5$

```

Q2. Demonstrate creation, writing to and reading from a pipe.

```
#include <sys/wait.h>
#include <assert.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

int main(int argc, char *argv[])
{
    int pfd[2];
    pid_t cpid;
    char buf;
    assert(argc == 2);
    if (pipe(pfd) == -1)
    {
        perror("pipe");
        exit(EXIT_FAILURE);
    }
    cpid = fork();
    if (cpid == -1)
    {
        perror("fork");
        exit(EXIT_FAILURE);
    }
    if (cpid == 0)
    {
        close(pfd[1]);
        while (read(pfd[0], &buf, 1) > 0)
            write(STDOUT_FILENO, &buf, 1);
        write(STDOUT_FILENO, "\n", 1);
        close(pfd[0]);
        exit(EXIT_SUCCESS);
    }
    else
    {
        close(pfd[0]);
        write(pfd[1], argv[1], strlen(argv[1]));
        close(pfd[1]);
        wait(NULL);
        exit(EXIT_SUCCESS);
    }
}
```



A terminal window titled "180905380@prg08: ~/Desktop/Operating Systems/Week 5" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
180905380@prg08:~/Desktop/Operating Systems/Week 5$ cc -o o2 pipe.c
180905380@prg08:~/Desktop/Operating Systems/Week 5$ ./o2 "Sample Input"
Sample Input
180905380@prg08:~/Desktop/Operating Systems/Week 5$
```

Q3. Write a C program to implement one side of FIFO.

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

#define FIFO_NAME "/tmp/my_fifo"
#define BUFFER_SIZE PIPE_BUF

int main()
{
    int pipe_fd;
    int res;
    int open_mode_1 = O_WRONLY;
    int open_mode_2 = O_RDONLY;
    int user_mode;
    int run = 1;
    printf("Enter 1 for write first, 2 for read first:\t");
    scanf("%d",&user_mode);
    if (user_mode == 1)
    {
        if (access(FIFO_NAME,F_OK) == -1)
        {
            res = mkfifo(FIFO_NAME, 0777);
            if (res != 0)
            {
                fprintf(stderr, "Could not create fifo %s\n", FIFO_NAME);
                exit(EXIT_FAILURE);
            }
        }
    }
    while(run)
    {
        if (user_mode == 1)
        {
            char buffer1[BUFFER_SIZE + 1];
            pipe_fd = open(FIFO_NAME, open_mode_1);
            printf(">> ");
            scanf("%s",buffer1);
            if (pipe_fd != -1)
            {
                res = write(pipe_fd, buffer1, BUFFER_SIZE);
                if (res == -1)
                {
                    fprintf(stderr, "Write error on pipe\n");
                    exit(EXIT_FAILURE);
                }
                close(pipe_fd);
                user_mode = 2;
                if (strcmp(buffer1,"quit") == 0)
                    run = 0;
            }
        }
        else
            exit(EXIT_FAILURE);
    }
    else if (user_mode == 2)
```

```

    {
        char buffer2[BUFFER_SIZE + 1];
        memset(buffer2, '\0', sizeof(buffer2));
        pipe_fd = open(FIFO_NAME, open_mode_2);
        if (pipe_fd != -1)
        {
            do
            {
                res = read(pipe_fd,buffer2,BUFFER_SIZE);
            }while (res > 0);
            close(pipe_fd);
            user_mode = 1;
            printf("%s\n",buffer2);
            if (strcmp(buffer2,"quit") == 0)
                run = 0;
        }
        else
            exit(EXIT_FAILURE);
    }
}
exit(EXIT_SUCCESS);
}

```

The image shows two terminal windows side-by-side, both titled "180905380@prg08: ~/Desktop/Operating Systems/Week 5".

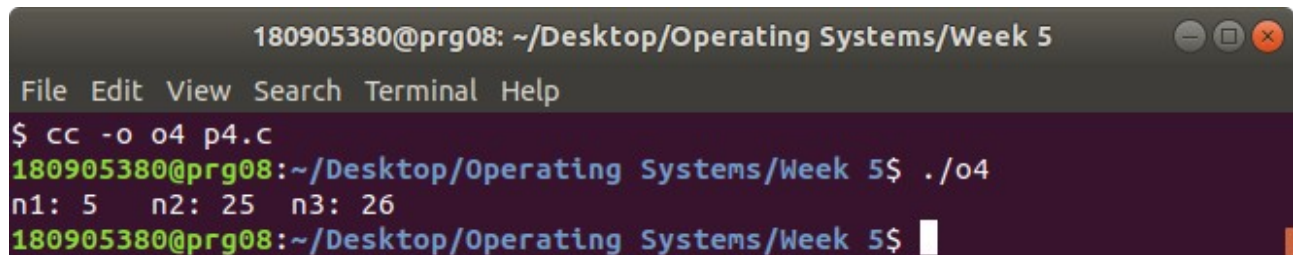
The left terminal window shows the command prompt "180905380@prg08:~/Desktop/Operating Systems/Week 5\$./o3". The user enters "1" for "write first". The program then displays a list of numbers from "one" to "ten", and the user types "quit".

The right terminal window shows the command prompt "180905380@prg08:~/Desktop/Operating Systems/Week 5\$./o3". The user enters "2" for "read first". The program then displays a list of numbers from "one" to "ten", and the user types "quit".

Q4. Write a C program reading and writing a binary files in C.

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

int main()
{
    int n = 5;
    int num[3];
    FILE *fptr;
    if ((fptr = fopen("hey.bin", "wb")) == NULL)
    {
        printf("Error! opening file");
        exit(1);
    }
    num[0] = n;
    num[1] = 5*n;
    num[2] = 5*n + 1;
    fwrite(num, sizeof(num), 1, fptr);
    fclose(fptr);
    if ((fptr = fopen("hey.bin", "rb")) == NULL)
    {
        printf("Error! opening file");
        exit(1);
    }
    fread(num, sizeof(num), 1, fptr);
    printf("n1: %d\tn2: %d\tn3: %d\n", num[0], num[1], num[2]);
    fclose(fptr);
    return 0;
}
```



A terminal window titled "180905380@prg08: ~/Desktop/Operating Systems/Week 5" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
$ cc -o o4 p4.c
180905380@prg08:~/Desktop/Operating Systems/Week 5$ ./o4
n1: 5    n2: 25   n3: 26
180905380@prg08:~/Desktop/Operating Systems/Week 5$
```