EX:5 INTERPROCESS COMMUNICATION

-S.Vishakan CSE-C 18 5001 196

SOURCE CODE – (Uppercase – Parent & Child):

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio_ext.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

char *upperCase(char *word);    //function for converting to uppercase

int main(void){
   int pid, id;
   char *l_word, *u_word;

   id = shmget(111, 1024, IPC_CREAT|00666);    //identifier for shared memory
   pid = fork();   //child process and parent process share the same memory

   if(pid > 0){
      l_word = (char*) shmat(id,(void*)0,0);  //attaching to shared memory
      printf("\nEnter a Word: ");
      fgets(l_word, 1000, stdin);
      wait(0);
      shmdt(l_word);  //detaching from shared memory after placing the word
   }

   else{
      sleep(7);   //waiting for 7 seconds when the writer process is putting words into
shared memory
      u_word = (char*) shmat(id,(void*)0,0);  //attaching once writer is complete
      printf("Received word : %s",upperCase(u_word));
```

```
        shmdt(u_word);  //detach
        exit(0);
    }

}


char *upperCase(char *word){
    int len = strlen(word);
    int i = 0;
    char *uword;
    uword = (char *)malloc(sizeof(char)*len);

    for(;i<len;i++)
        uword[i] = toupper(word[i]);

    return uword;
}
```

OUTPUT:

(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess Communication$ gcc 1-Uppercase.c -o u
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess Communication$ ./u

Enter a Word: interprocess communication
Received word : INTERPROCESS COMMUNICATION

SOURCE CODE – (Server Program):

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio_ext.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void){
    int id;
    char *msg;
    char *contents = (char *)malloc(1024);
    char *buf = (char *)malloc(255);
    FILE *src;

    id = shmget(111, 1024, IPC_CREAT|00666);    //getting ID of shared mem.
    msg = shmat(id, NULL, 0);                   //msg var. is now a shared mem. var
    printf("\nReceived the following file name to be opened: %s\n",msg);

    src = fopen(msg, "r");      //opening the file that came thro' the client
    printf("\nFile Opening...");

    if(src == NULL){    //if file is not available
            strcpy(msg, "File Not Available");
       printf("\nProcess finished execution with File Not Found error.\n");
       exit(0);
    }

    while(fgets(buf, 255, src) != NULL){    //copying contents to a temp. var. using
fgets()
       strcat(contents, buf);
    };

    fclose(src);
    strcpy(msg, contents);      //copying contents to shared mem. var.
    shmdt(msg);
    printf("\nProcess finished execution without errors.\n");
    exit(0);
}
```

OUTPUT:

vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcen$ gcc 2-Server.c -o s
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcess Communication$ ./s

Received the following file name to be opened: sample.txt

File Opening...
Process finished execution.

(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcess Communication$ ./s

Received the following file name to be opened: samp.txt

File Opening...
Process finished execution with File Not Found error.

SOURCE CODE – (Client Program):

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio_ext.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void){
    int id;
    char *msg;

    id = shmget(111, 1024, IPC_CREAT|00666);
    msg = shmat(id, NULL, 0);
    printf("Enter the File to be Transferred: ");
    scanf("%[^\n]",msg);
```

```
    sleep(8);       //Waiting for Server to read the specified file and send output

    printf("\nContents of File: \n");
    printf("%s\n",msg);
    shmdt(msg);

    shmctl(id, IPC_RMID, NULL); //destroying the shared memory & contents
    exit(0);

}
```

OUTPUT:

(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcess Communication$ gcc 2-Client.c -o c
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcess Communication$ ./c
Enter the File to be Transferred: sample.txt

Contents of File:
Hi
This is Vikram V of CSE-C
Today is February 14!

(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex5 InterProcess Communication$ ./c
Enter the File to be Transferred: samp.txt

Contents of File:
File Not Available

SOURCE CODE – (Peer 1 Program):

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio_ext.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void){
    int id;
    char msg[100], *buffer;
    char temp1[100], temp2[100];

    printf("\n\t\t\t\tP2P Chat\n\n");

    id = shmget(111, 1024, IPC_CREAT|00666);    //Opening a shared memory access
    buffer = shmat(id, NULL, 0);                //Attaching a buffer to it

    printf("\n\n\t\tThe Chat Connection has been Opened!.\n\t\t\tEnter \"Bye\" to Quit.\
n");

    printf("\nYou:\n\t");
    fgets(temp1, 100, stdin);

    strcpy(msg, "~");             //Clearing the strings with a preset ~ value
    strcat(msg, temp1);
    strcpy(buffer, msg);
    strcpy(msg, "~");


    while(1){
        strcpy(msg, "~");                 //Clearing the msg buffer with ~ again for next
time
        while(buffer[0] == '~');          //Waiting, if the buffer is empty.
        strcpy(temp2, buffer);
        char *sep = strtok(temp2, "`");     //Splitting the string at the preset value ` for
the other client

        printf("\nPeer:\n\t%s\n", sep);
```

```
        if(strcmp(sep, "Bye\n") == 0){     //Ending the chat if "Bye" is entered by the other user.
            break;
        }

        else{
            printf("You:\n\t");
            fgets(temp1, 100, stdin);
            strcat(msg, temp1);             //Putting the scanned value into buffer
            strcpy(buffer, msg);            //Now buffer is like ~<msg>
            strcat(msg, "~");               //Now msg is like ~<msg>~

            if(strcmp(temp1, "Bye\n") == 0){    //Exiting the chat this user enters "Bye"
                break;
            }
        }

    }

    printf("\n\n\t\tThe Chat Connection has been Closed.\n");
    shmdt(buffer);
    shmctl(id, IPC_RMID, NULL);     //Deleting the shared memory addressing
    sleep(0);
    exit(0);

}
```

OUTPUT:

(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess Communication$ gcc 3-ChatPeer1.c -o c1
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess Communication$ ./c1

P2P Chat

The Chat Connection has been Opened!.
Enter "Bye" to Quit.

You:
Hey

Peer:
Hello

You:

How's it going?

Peer:

It's going good!

You:

Good to know :)

Peer:

Bye

You:

Bye

The Chat Connection has been Closed.

SOURCE CODE – (Peer 2 Program):

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio_ext.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void){
    int id;
    char msg[100], *buffer;
    char temp1[100], temp2[100];

    printf("\n\t\t\t\tP2P Chat\n\n");

    id = shmget(111, 1024, IPC_CREAT|00666);    //Opening a shared memory access
    buffer = shmat(id, NULL, 0);                //Attaching a buffer to it

    printf("\n\n\t\tThe Chat Connection has been Opened!.\n\t\t\tEnter \"Bye\" to Quit.\
n");
```

```
    strcpy(msg, "`");       //Clearing the strings with a preset ` value
    strcpy(buffer, "`");

    while(1){
        strcpy(msg, "`");                   //Clearing the msg buffer with ` again for next
time
        while(buffer[0] == '`');            //Waiting, if the buffer is empty.
        strcpy(temp2, buffer);
        char *sep = strtok(temp2, "~");     //Splitting the string at the preset value ~ for
the other client

        printf("\nPeer:\n\t%s\n", sep);

        if(strcmp(sep, "Bye\n") == 0){      //Ending the chat if "Bye" is entered by the
other user.
            break;
        }

        else{
            printf("You:\n\t");
            fgets(temp1, 100, stdin);
            strcat(msg, temp1);             //Putting the scanned value into buffer
            strcpy(buffer, msg);            //Now buffer is like `<msg>
            strcat(msg, "`");               //Now msg is like `<msg>`

            if(strcmp(temp1, "Bye\n") == 0){    //Exiting the chat this user enters "Bye"
                break;
            }
        }

    }

    printf("\n\n\t\tThe Chat Connection has been Closed.\n");
    shmdt(buffer);
    shmctl(id, IPC_RMID, NULL);     //Deleting the shared memory addressing
    sleep(0);
    exit(0);

}
```

OUTPUT:

```
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess
Communication$ gcc 3-ChatPeer2.c -o c2
(base) vishakan@Legion:~/Desktop/Operating-Systems/Ex4 InterProcess
Communication$ ./c2

                    P2P Chat


            The Chat Connection has been Opened!.
                Enter "Bye" to Quit.

Peer:
        Hey

You:
        Hello

Peer:
        How's it going?

You:
        It's going good!

Peer:
        Good to know :)

You:
        Bye


            The Chat Connection has been Closed.
```