**Assignment Number 03**

**Name:** Mihir Unmesh Patil

**Roll NO**: TYCOC213

**Batch:** C/C-3

**CODE:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

void process(int id, int read\_fd, int write\_fd) {

char buffer[100];

while (1) {

// Read message

read(read\_fd, buffer, sizeof(buffer));

if (strncmp(buffer, "exit", 4) == 0 || strncmp(buffer, "stop", 4) == 0) {

printf("Process %d received exit command.\n", id);

break;

}

printf("Process %d received: %s\n", id, buffer);

// Send message

printf("Process %d, enter message: ", id);

fgets(buffer, sizeof(buffer), stdin);

write(write\_fd, buffer, strlen(buffer) + 1);

if (strncmp(buffer, "exit", 4) == 0 || strncmp(buffer, "stop", 4) == 0) {

printf("Process %d sent exit command.\n", id);

break;

}

}

close(read\_fd);

close(write\_fd);

exit(0);

}

void communication() {

int pipe\_fd1[2], pipe\_fd2[2];

char buffer[100];

if (pipe(pipe\_fd1) == -1 || pipe(pipe\_fd2) == -1) {

perror("Pipe failed");

exit(1);

}

pid\_t pid = fork();

if (pid > 0) {

// Process 1

close(pipe\_fd1[0]); // Close unused read end of pipe1

close(pipe\_fd2[1]); // Close unused write end of pipe2

while (1) {

printf("Process 1, enter message: ");

fgets(buffer, sizeof(buffer), stdin);

write(pipe\_fd1[1], buffer, strlen(buffer) + 1);

if (strncmp(buffer, "exit", 4) == 0 || strncmp(buffer, "stop", 4) == 0) {

printf("Process 1 sent exit command.\n");

break;

}

read(pipe\_fd2[0], buffer, sizeof(buffer));

if (strncmp(buffer, "exit", 4) == 0 || strncmp(buffer, "stop", 4) == 0) {

printf("Process 1 received exit command.\n");

break;

}

printf("Process 1 received: %s\n", buffer);

}

close(pipe\_fd1[1]);

close(pipe\_fd2[0]);

} else if (pid == 0) {

// Process 2

close(pipe\_fd1[1]); // Close unused write end of pipe1

close(pipe\_fd2[0]); // Close unused read end of pipe2

process(2, pipe\_fd1[0], pipe\_fd2[1]);

} else {

perror("Fork failed");

exit(1);

}

}

int main() {

communication();

return 0;

}

**Output:**

