1. **FIFOS:** Full duplex communication between two independent processes. First process accepts sentences and writes on one pipe to be read by second process and second process counts number of characters, number of words and number of lines in accepted sentences, writes this output in a text file and writes the contents of the file on second pipe to be read by first process and displays on standard output.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

int main(int argc, char const \*argv[]) {

int fd[2], fd1[2]; // File Descriptors for the Pipes

pid\_t pid;

FILE \*fp;

if (pipe(fd1) == -1 || pipe(fd) == -1) {

perror("Error occured while creating Pipes\n");

exit(-1);

}

// Fork call to create Child Process

pid = fork();

// Checking if Child Process is created Successfully

if (pid == -1) {

perror("Error occured while creating Child Process\n");

exit(-1);

} else if (pid == 0) {

// This is the Child Process

close(fd[1]); // Closing writer part of the pipe, since Child Process only reads from First Pipe

char pathName[100];

// Reading the Pathname written by the Parent Process

read(fd[0], pathName, sizeof(char)\* 100);

// Creating File at Path Name and checking if File is Successfully created

if ((fp = fopen(pathName, "w")) == NULL) {

perror("Error occured while creating file\n");

exit(-1);

}

char mString[] = "Pipes: Full duplex communication between parent and child processes";

// Printing String onto File

fprintf(fp, "%s", mString);

// Closing unused Reader Part of Pipe 2

close(fd1[0]);

// Writing File Pointer on the Writer part of Pipe 2

write(fd1[1], fp, sizeof(fp));

// Close Writer Part of Pipe 2

close(fd1[1]);

} else {

// This is the Parent Process

// Since Parent Process is to write Pathname of the File, we close the unused reader part of the pipe

close(fd[0]); // The Reader Part of the Pipe corresponds to 0

char \*pathName = "temp.txt";

// Writing pathName on the Pipe

write(fd[1], pathName, strlen(pathName));

close(fd[1]); // Since Write Operation has been performed, close the Pipe

wait(NULL); // Waiting for Child Process to complete its Execution.

// Closing unused Writer Part of Pipe 2

close(fd1[1]);

// read(fd1[1], \*fp, 4096);

char mString[100];

//Scanning String from file

fscanf(fp, "%s", mString);

printf("%s\n", mString);

fclose(fp);

}

return 0;

}