**CODE MẪU UNNAME PIPE – NAME PIPE**

**Unname pipe:**

#include <stdio.h>

#include <unistd.h>

#include <string.h>

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

{

int fp1[2],fp2[2];

int buffer;

int pid;

if(argc<2) {

printf("Doi so thieu.\n");

return -1;

}

if(pipe(fp1)==0&&pipe(fp2)==0) {

pid = fork();

if(pid<0) {printf("Fork failed\n"); return -1;}

else if(pid==0) {

close(fp1[1]);

read(fp1[0], &buffer, sizeof(buffer));

printf("Read from parents: %d\n", buffer);

close(fp1[0]);

int n=10+buffer;

printf("data send to parent: %d \n",n);

close(fp2[0]);

write(fp2[1], &n, sizeof(n));

close(fp2[1]);

}

else {

close(fp1[0]);

printf("Data from parents: %s\n", argv[1]);

int temp =atoi(argv[1]);

write(fp1[1], &temp, sizeof(temp));

close(fp1[1]);

printf("da viet \n");

close(fp2[1]);

int tam;

read(fp2[0], &tam,sizeof(tam));

printf("data get from child %d \n",tam);

close(fp2[0]);

}

}

else {printf("Pipe failed\n"); return -2;}

}

**Name Pipe:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <sys/errno.h>

#define FIFO1 "/tmp/ff.1"

#define FIFO2 "/tmp/ff.2"

#define PM 0666

extern int errno;

#define PIPE\_BUF 4096

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

{

char s1[PIPE\_BUF], s2[PIPE\_BUF];

int childpid, readfd, writefd;

if((mknod(FIFO1, S\_IFIFO | PM, 0)<0)&&(errno!=EEXIST)){

printf("Fail to create FIFO 1. Aborted.\n");

return -1;

}

if((mknod(FIFO2, S\_IFIFO | PM, 0)<0)&&(errno!=EEXIST)){

unlink(FIFO1);

printf("Fail to create FIFO 2. Aborted.\n");

return -1;

}

childpid=fork();

if(childpid==0){ //child

if((readfd=open(FIFO1, 0))<0)

perror("Child cannot open readFIFO.\n");

if((writefd=open(FIFO2, 1))<0)

perror("Child cannot open writeFIFO.\n");

read(readfd, s2, PIPE\_BUF);

printf("Child read from parent: %s\n", s2);

printf("Enter response: ");

gets(s1);

write(writefd, s1, strlen(s1));

close(readfd);

close(writefd);

return 1;

}

else if(childpid>0) { //parent

if((writefd=open(FIFO1, 1))<0)

perror("Parent cannot open writeFIFO.\n");

if((readfd=open(FIFO2, 0))<0)

perror("Child cannot open readFIFO.\n");

printf("Enter data to FIFO1: ");

gets(s1);

write(writefd, s1, strlen(s1));

read(readfd, s2, PIPE\_BUF);

printf("Parent read from child: %s\n", s2);

while(wait((int\*) 0)!=childpid);

close(readfd);

close(writefd);

if(unlink(FIFO1)<0)

perror("Cannot remove FIFO1.\n");

if(unlink(FIFO2)<0)

perror("Cannot remove FIFO2.\n");

return 1;

}

else { printf("Fork failed\n"); return -1;}

}