姓名：吴涵 学号：202101000720 成绩：

# 实验30管道通信

1、匿名管道通信

源程序：

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

int wc = 1, rc = 1;

void writer(const char \*message, int count, FILE \*stream)

{

    for (; count > 0; --count)

    {

        fprintf(stream, "%s:%d\n", message, wc);

        sprintf(wstr, "%s:%d\n", message, wc);

        fflush(stream);

        printf("第%d次写入管道的内容为：%s", wc, wstr);

        wc++;

        sleep(1);

    }

}

void reader(FILE \*stream)

{ // 读管道stream，次数为rc

    char buffer[1024];

    while (!feof(stream) && !ferror(stream) && fgets(buffer, sizeof(buffer), stream) != NULL)

    { // 从管道stream读取消息存入buffer缓冲区

        printf("读进程第%d次读取：", rc);

        rc++;

        fputs(buffer, stdout); // 输出所读管道信息

    }

}

int main()

{

    int fds[2];

    pid\_t pid;

    pipe(fds);    // 创建匿名管道

    pid = fork(); // 创建子进程

    if (pid == (pid\_t)0)

    { // 子进程代码

        FILE \*stream;

        close(fds[1]);                // 关闭管道写端

        stream = fdopen(fds[0], "r"); // 子进程以只读方式打开管道

        reader(stream);               // 子进程读取管道

        close(fds[0]);                // 子进程关闭读端

    }

    else

    { // 父进程代码

        FILE \*stream;

        close(fds[0]);                      // 关闭管道读端

        stream = fdopen(fds[1], "w");       // 父进程以只写方式打开管道

        writer("Hello, world.", 5, stream); // 父进程向管道写入5次消息“Hello, world.”

        close(fds[1]);                      // 子进程关闭写端

    }

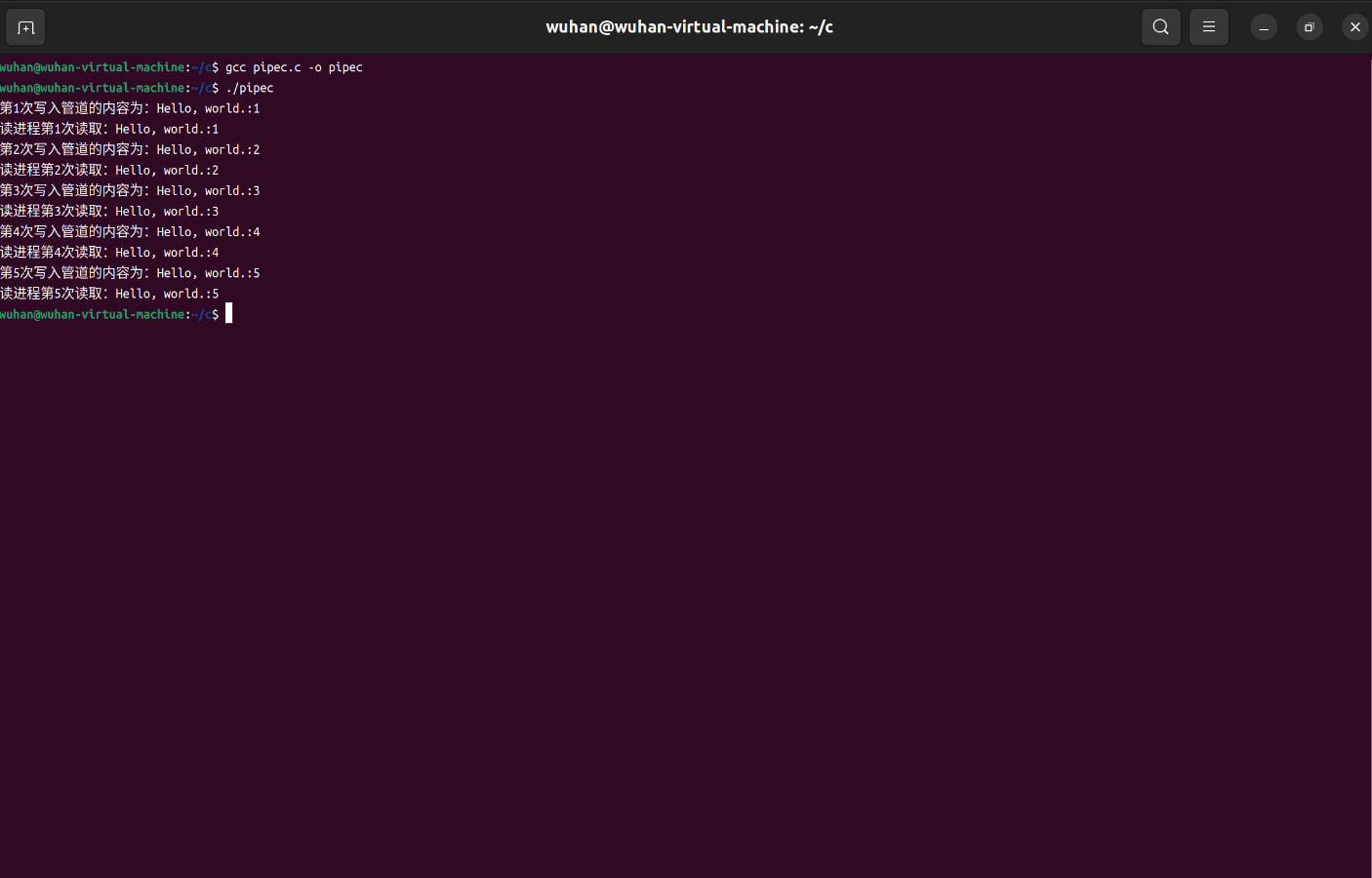
    return 0;

}

编译链接命令：gcc pipec.c -o pipec

运行命令：./pipec

交互与结果：



2、有名管道通信

源程序：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <errno.h>

#include <unistd.h>

#include <fcntl.h>

#define FIFO "myfifo" // 有名管道名字

#define BUFF\_SIZE 1024

int main()

{

char buff[BUFF\_SIZE]; // 欲读取管道的数据缓冲区

int real\_read; // 读取管道的字节数

int fd; // 管道描述符

int rc = 1; // 管道读次数

if (access(FIFO, F\_OK) == -1)

{ // 测试有名管道FIFO是否存在，若不存在，则用mkfifo创建该管道

if ((mkfifo(FIFO, 0666) < 0) && (errno != EEXIST))

{ // 创建管道"myfifo"，允许读写

printf("Can NOT create fifo file!\n");

exit(1);

}

}

if ((fd = open(FIFO, O\_RDONLY)) == -1)

{ // 以只读方式打开FIFO，返回文件描述符fd

printf("Open fifo error!\n");

exit(1);

}

while (1)

{ // 循环读管道，若读空，则结束循环

memset(buff, 0, BUFF\_SIZE);

if ((real\_read = read(fd, buff, BUFF\_SIZE)) > 0)

printf("第%d次读取管道: '%s'.\n", rc++, buff);

else

break;

}

close(fd);

exit(0);

}

编译链接命令：

gcc fifo\_write.c -o fifo\_write

gcc fifo\_read.c -o fifo\_read

运行命令：

./fifo\_write

./fifo\_read

交互与结果：

