四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

学号：2022141460176 姓名：杨一舟 专业：计算机科学与技术 第 9 周

课程名称	操作系统课程设计	实验课时	2
实验项目	进程线程与管道通信	实验时间	2023 年 4 月 22 日
实验目的	利用进程通信方法，实现简单的划拳猜数字游戏		
实验环境	Ubuntu-Linux 操作系统、 gcc 编译工具		
实验内容（算法、程序、步骤和方法）	**1.实验原理；** 命名管道由 **mkfifo** 函数创建，使用 **open** 函数打开 命名管道打开规则： 以只读的方式打开 FIFO 文件 O_NONBLOCK disable：阻塞直到有相应进程以写的方式打开 FIFO 文件 O_NONBLOCK enable：立刻返回成功 以只写的方式打开 FIFO 文件 O_NONBLOCK disable：阻塞直到有相应进程以读的方式打开 FIFO 文件 O_NONBLOCK enable：打开失败并且立刻返回，错误码为 ENXIO **2.游戏实现过程：** （1）利用管道通信的方式，实现玩家进程和裁判进程之间的通信，命名管道 fifo 作为出拳通道，result 管道作为获取游戏结果通道。		

```
mountain@Lumous:~$ mkfifo fifo
```

```
mountain@Lumous:~$ mkfifo result
```

（2）编写 players.c 和 server.c 代码(在 Dev-C++中展示)

Players.c

```c
#include <unistd.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <string.h>
#define MAX_BUF 1024

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Usage: %s player_id\n", argv[0]);
        exit(1);
    }

    char *player_id = argv[1];
    char buf[MAX_BUF];
    int fifo_fd, result_fd;
    int round_num, fist_num, guess_sum;

   // 打开fifo管道和result管道
    fifo_fd = open("fifo", O_WRONLY);
    result_fd = open("result", O_RDONLY);

   // 获取用户输入
    printf("Round: ");
    scanf("%d", &round_num);
    printf("Fist: ");
    scanf("%d", &fist_num);
    printf("Guess sum: ");
    scanf("%d", &guess_sum);

   // 将数据写入fifo管道
    sprintf(buf, "%s,%d,%d,%d", player_id, round_num, fist_num, guess_sum);
    write(fifo_fd, buf, strlen(buf) + 1);

  // 从result管道读取裁判进程返回的结果

    read(result_fd, buf, MAX_BUF);


    // 关闭管道
    close(fifo_fd);
    close(result_fd);

    return 0;
}
```

Server.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <string.h>
#define MAX_BUF 1024

int main() {
    char buf[MAX_BUF];
    int fifo_fd, result_fd;
    int p1_round_num, p1_fist_num, p1_guess_sum;
    int p2_round_num, p2_fist_num, p2_guess_sum;
    int sum;

    // 打开fifo管道和result管道
    fifo_fd = open("fifo", O_RDONLY);
    result_fd = open("result", O_WRONLY);
// 从fifo管道读取两个玩家的数据
read(fifo_fd, buf, MAX_BUF);
sscanf(buf, "%*[^,],%d,%d,%d", &p1_round_num, &p1_fist_num, &p1_guess_sum);

read(fifo_fd, buf, MAX_BUF);
sscanf(buf, "%*[^,],%d,%d,%d", &p2_round_num, &p2_fist_num, &p2_guess_sum);

// 计算两个玩家的出拳数字和
sum = p1_fist_num + p2_fist_num;
// 判断哪个玩家猜的数字总和与计算结果一致
if (p1_guess_sum == sum && p2_guess_sum != sum)
{
    sprintf(buf, "Round %d: Player 1 wins!", p1_round_num);
    printf("Round %d: Player 1 wins!", p1_round_num);
    printf("\n");
}
else if (p1_guess_sum != sum && p2_guess_sum == sum) {
    sprintf(buf, "Round %d: Player 2 wins!", p1_round_num);
    printf("Round %d: Player 2 wins!", p1_round_num);
    printf("\n");

} else if(p1_guess_sum != sum && p2_guess_sum != sum)
{
    sprintf(buf, "Round %d: no winner!", p1_round_num);
    printf("Round %d: no winner!", p1_round_num);
    printf("\n");
}



else {
    sprintf(buf, "Round %d: both winner!", p1_round_num);
    printf("Round %d: both winner!", p1_round_num);
    printf("\n");
}


//将结果通过result管道返回给玩家进程
  write(result_fd, buf, strlen(buf) + 1);


// 关闭管道
close(fifo_fd);
close(result_fd);

return 0;
}
```

（3）利用 gcc 工具对 c 语言代码进行编译连接

“gcc filename -o outfilename”利用“-o”命令可以指定所输出的文件名称

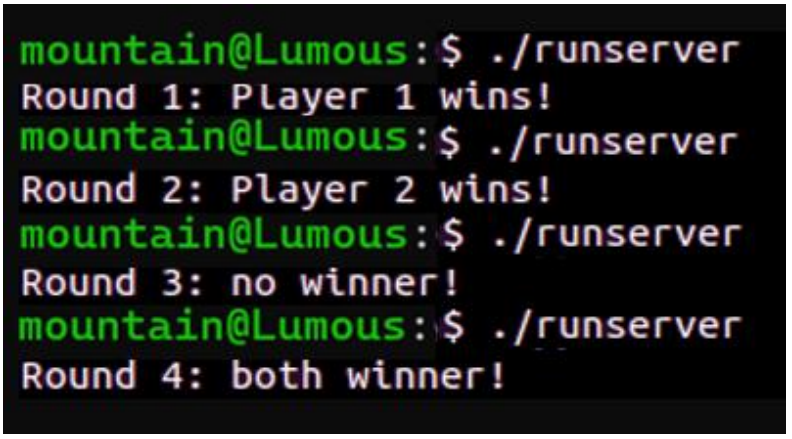
（4）利用./filename 命令运行 runplayers 和 runserver

（5）开启三个终端，分别为玩家 p1，玩家 p2，裁判 server

玩家进程输入：轮次 出拳数字 所猜数字

```
mountain@Lumous:$ gcc players.c -o runplayers
mountain@Lumous:$ ./runplayers p1
Round: 1
Fist: 2
Guess sum: 5
mountain@Lumous:$ ./runplayers p1
Round: 2
Fist: 0
Guess sum: 3
mountain@Lumous:$ ./runplayers p1
Round: 3
Fist: 4
Guess sum: 7
mountain@Lumous:$ ./runplayers p1
Round: 4
Fist: 2
Guess sum: 4
```

```
mountain@Lumous:$ ./runplayers p2
Round: 1
Fist: 3
Guess sum: 6
mountain@Lumous:$ ./runplayers p2
Round: 2
Fist: 4
Guess sum: 4
mountain@Lumous:$ ./runplayers p2
Round: 3
Fist: 2
Guess sum: 3
mountain@Lumous:$ ./runplayers p2
Round: 4
Fist: 2
Guess sum: 4
```

	裁判进程判断游戏结果： 
数据记录和计算	各进程运行结果如过程记录所示
结　论	通过管道通信编程，成功实现了划拳猜数字小游戏 Player 发送参数到管道 fifo 并从管道 result 中读取结果 Server 从管道 fifo 接收参数判断输赢并发送结果到管道 result
小结	通过本次实验，我深入理解了命名管道在进程间通信中的应用。通过实践，我掌握了命名管道的创建、打开、读写及关闭等操作，实现了进程间的数据交换。
指导老师评议	成绩评定：　　指导教师签名：