**西南石油大学实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **项目编号** | **实验二** | **项目名称** | **进程间的通信** | **成绩** |  |
| **专业年级** | **软件工程2022级** | | | **指导教师** | **刘义军** |
| **姓名** | **李浩楠** | **学号** | **202231060920** | **实验日期** | **2023/11/11** |

1. **实验目的**

**实现进程间的通信，加深对进程间通信的理解，为以后的程序开发之路夯实基础。**

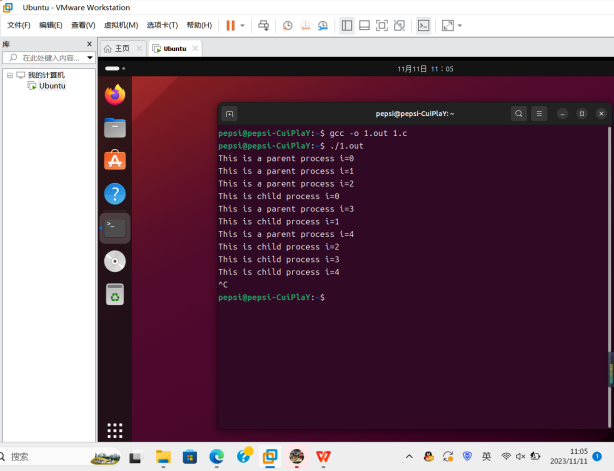
1. **实验工具**

VMware Workstation Ubuntu

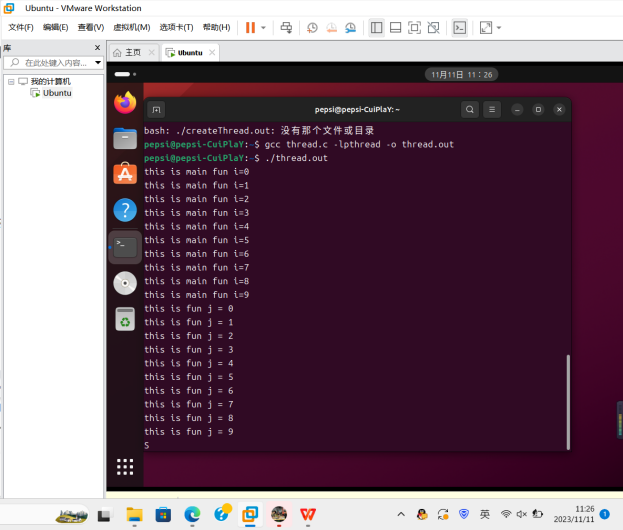
1. **实验步骤**

在Linux系统上编写代码，保存后编译并执行，输入相关的测试用例，记录测试结果

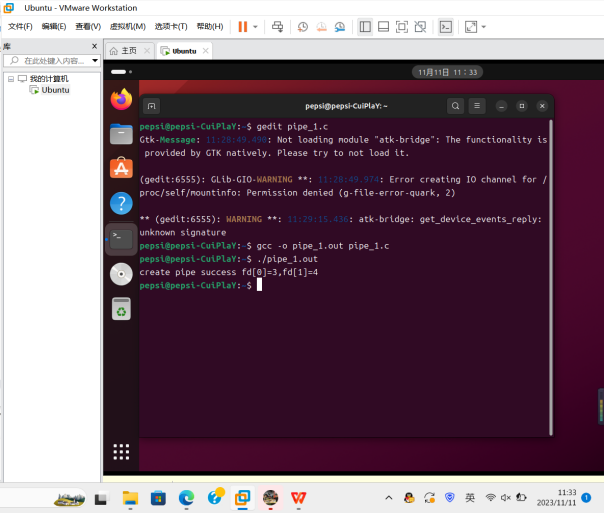
1. **实验结果**
2. 进程间的通信



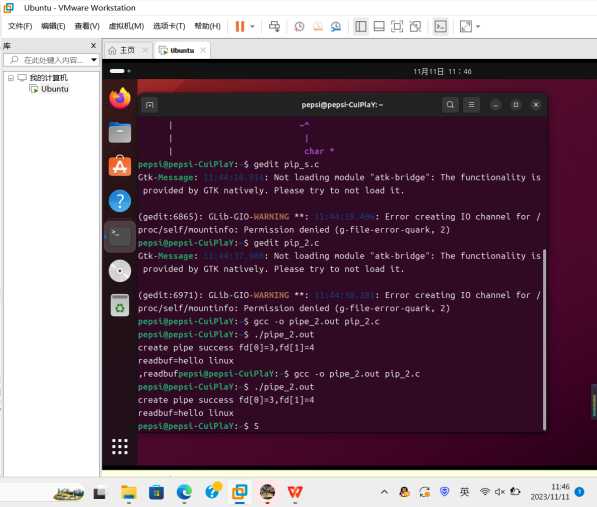
1. 线程间的通信



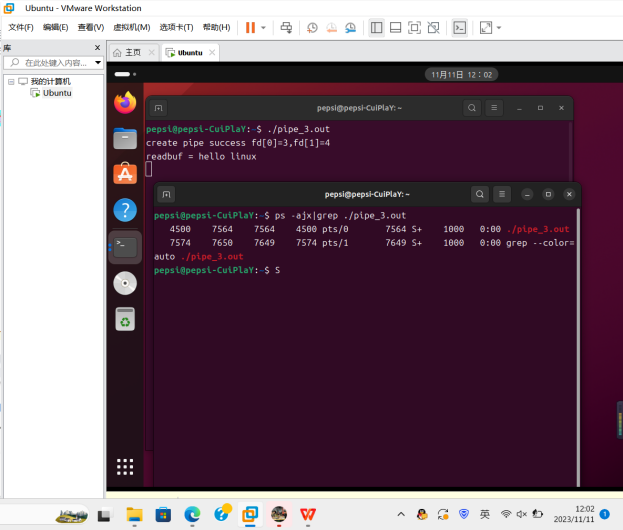
1. 无名管道通信——pipe函数



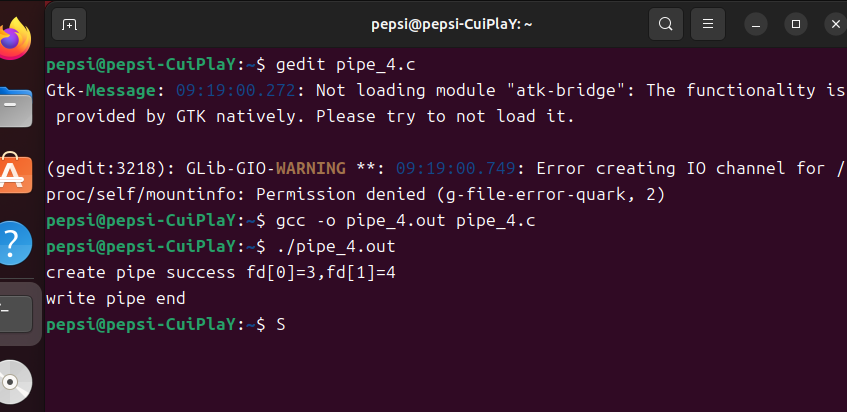
1. 单进程的内部的读写



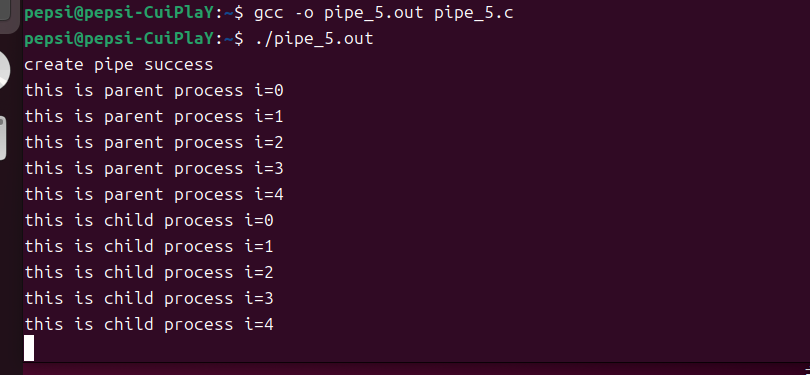
1. 读阻塞



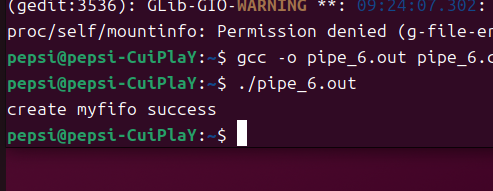
1. 写阻塞



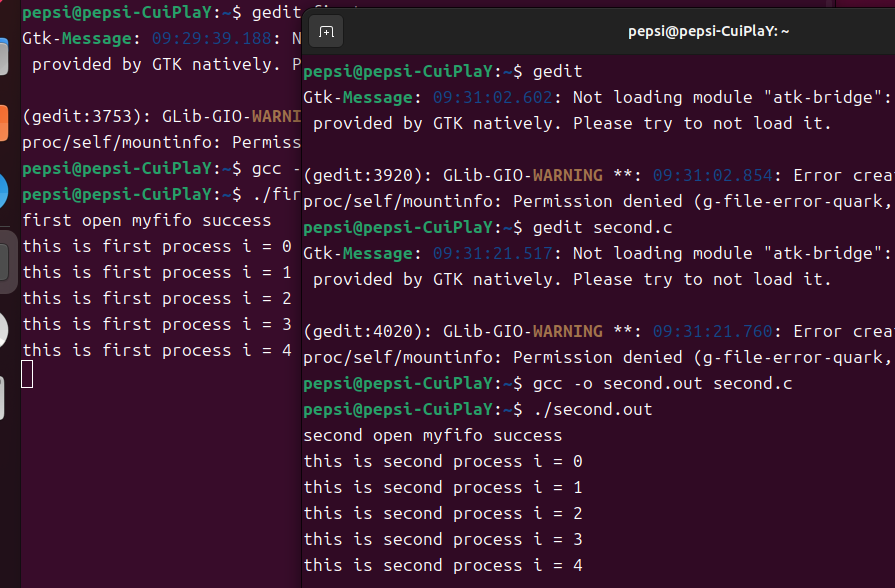
1. 综合实例1



1. 创建管道文件



1. 综合实验2



1. **实验总结**

1.进程间的通信

错误分析：在父子进程中执行的代码输出的语句相似，可能会导致混淆，通常子进程和父进程执行不同的任务。

问题分析：程序陷入死循环。在实际的程序中，我们应该避免使用无限循环，这可能会导致程序无法正常结束。

1. 线程间的通信

在fun中，while循环的条件是thread\_inter == 0，这意味着只有当thread\_inter为0时，fun函数才会继续执行。然而，在main函数中，thread\_inter被设置为1，会陷入死循环。

1. 无名管道通信——pipe函数

代码创建了一个管道，并打印出管道的fd[0]和fd[1]。如果创建管道成功，将打印create pipe success fd[0]=x,fd[1]=y。如果创建管道失败，将打印"create pipe failure"。

1. 单进程的内部的读写

代码创建了一个管道，并将字符串"hello linux"写入管道。然后从管道中读取数据，并将其打印出来。最后关闭了管道的两个文件。如果创建管道失败，将返回-1并打印错误消息。

1. 读阻塞

结果分析： 创建了一个管道，将字符串"hello linux"写入管道。从管道中读取数据，并将其打印出来。清空readbuf数组，准备第二次从管道中读取数据。由于第一次读取后没有关闭管道的写端，导致第二次读取时被阻塞。。

错误分析和问题分析：

第一次读取后没有关闭管道的写端，导致第二次读取时被阻塞。应该在第一次读取后关闭管道的写端。第二次读取时，由于管道已经被关闭，会导致读取失败。应该在第二次读取前重新打开管道的读端。第二次读取后，应该再次清空readbuf数组，以便下一次读取。

1. 写阻塞

代码创建了一个管道，并将字符串"hello linux"写入管道。然后关闭了管道的读端，等待另一个进程从管道中读取数据。由于没有其他进程来读取数据，所以程序会一直运行下去，直到手动停止。

错误分析和问题分析：

在循环结束后，应该再次打开管道的读端，以便另一个进程可以从管道中读取数据。为了避免无限循环，可以在循环内部添加一个条件来判断是否已经写入了足够的数据。

1. 综合实例1

代码创建了一个管道，并在父进程中打印出0到4的数字，每打印一次后暂停100微秒。然后，将process\_inter变量设置为1，并等待5秒。接着，子进程开始执行，读取管道中的数据，如果管道为空则暂停。当父进程修改了process\_inter变量并将其写入管道后，子进程会继续执行，打印出0到4的数字，每打印一次后暂停100微秒。

错误分析和问题分析：

在fork()函数调用后，没有对返回值进行判断，可能导致创建子进程失败。

在父进程中，将process\_inter变量设置为1后，没有立即将其写入管道。

在子进程中，读取管道中的数据时，没有检查读取的字节数是否正确。

1. 创建管道文件

该代码是一个C语言程序，用于创建一个名为"myfifo"的命名管道。程序首先包含了所需 的头 文件，然后在main函数中调用mkfifo函数来创建命名管道。如果创建成功，程序会 输出"create myfifo success"，否则输出"creat myfifo failure"并返回-1。

1. 综合实验2

第一个程序（first.c）：

1. 在main函数中，定义了一个整数变量fd用于存储文件，一个整数变量i用于循环计数，以及一个字符变量process\_inter用于存储进程间的交互。
2. 使用open函数打开名为"./myfifo"的命名管道，以写入模式（O\_WRONLY）。如果打开失败，打印错误信息并返回-1。
3. 如果打开成功，打印成功信息。
4. 使用for循环打印当前进程的编号，并暂停100微秒。
5. 将process\_inter设置为1，然后暂停5秒。
6. 使用write函数将process\_inter的值写入命名管道。

第二个程序（second.c）：

1. 在main函数中，定义了一个整数变量fd用于存储文件，一个整数变量i用于循环计数，以及一个字符变量process\_inter用于存储进程间的交互。
2. 使用open函数打开名为"./myfifo"的命名管道，以只读模式（O\_RDONLY）。如果打开失败，打印错误信息并返回-1。
3. 如果打开成功，打印成功信息。
4. 使用for循环打印当前进程的编号，并暂停100微秒。
5. 使用read函数从命名管道中读取process\_inter的值。
6. 检查process\_inter的值是否为1，如果是，则跳出循环。否则，继续等待其他进程发送数据。
7. 等待其他进程发送数据。
8. 返回0表示程序正常结束。

两个程序通过创建命名管道实现了进程间的通信。第一个作为写入方，将数据写入命名管道；第二个作为读取方，从管道中读取数据。当读取到数据时，程序会跳出循环并继续执行。