**实验二 进程同步互斥**

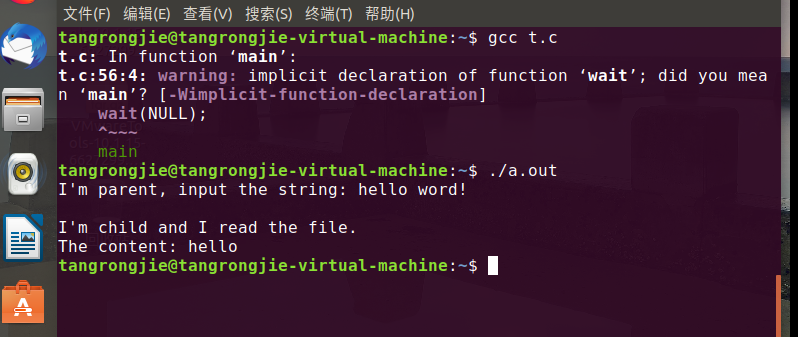
一 实验内容

设计并编写程序。父进程从外界获取字符串，并将其写入共享文件；子进程从共享文件中获取字符串，并将其打印出来。

二 实验程序代码

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #include<string.h>  #include<unistd.h>  #include<sys/types.h>  #include<sys/stat.h>  #include<fcntl.h>  //错误处理  int fatal(const char \*info)  {  perror(info);  exit(1);  }  //文件加锁  int lock(int f)  {  //定位到文件开头  lseek(f, 0, SEEK\_SET);  //加锁错误时给予提示  if(lockf(f, F\_LOCK, 0) == -1)  fatal("lockf()");  return 0;  }  //文件解锁  int unlock(int f)  {  lseek(f, 0, SEEK\_SET);  if(lockf(f, F\_ULOCK, 0) == -1)  fatal("unlockf()");  return 0;  }  int main()  {  int file, pid;  char str[50];    //当文件打开错误给予提示信息  if((file = open("df.txt", O\_RDWR|O\_CREAT|O\_TRUNC, 0744)) < 0)  fatal("open");  //创建子进程  while((pid = fork()) == -1);  //对子父进程分别做不同处理  switch(pid)  {  //错误时提示  case -1: fatal("Fork error!"); break;  //子进程读取文件内容  case 0:  //首先加锁，占有资源  lock(file);  //读取文件内容，长度为str数组中的字节数  read(file, str, sizeof(str));  //解锁文件  unlock(file);  //输出读取的内容  printf("\nI'm child and I read the file.\nThe content: %s\n", str);  exit(0);  //父进程读取输入并写入文件  default:  lock(file);  printf("I'm parent, input the string: ");  //输入内容  scanf("%s", str);  //写入文件  write(file, str, sizeof(str));  unlock(file);  //关闭文件  close(file);  //等待子进程完成读取  wait(NULL);  }  return 0;  } |

三 程序结果展示



【分析】：

从结果图中可看出，父进程首先执行并要求输入内容，当父进程将内容写入文件之后，子进程写入的文件进行读取，并将其输出到了终端上。从结果中，可看出子进程进行正确而定执行。