山东大学 计算机科学与技术 学院

操作系统 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201705130120 | 姓名：苑宗鹤 | | 班级： 1班 |
| 实验题目： 线程和管道通信实验 | | | |
| 实验学时：2 | | 实验日期：2019/12/22 | |
| 实验目的：  通过 Linux 系统中线程和管道通信机制的实验，加深对于线程控制和管道通信 概念的理解，观察和体验并发进/线程间的通信和协作的效果 ，练习利用无名管道 进行进/线程程通信的编程和调试技术。 | | | |
| 硬件环境：  pc | | | |
| 软件环境：  Win10 clion  Linux ubuntu | | | |
| 实验内容与设计：  设有二元函数 f(x,y) = f(x) + f(y)  其中： f(x) = f(x-1) \* x (x >1)  f(x)=1 (x=1)  f(y) = f(y-1) + f(y-2) (y> 2)  f(y)=1 (y=1,2)  请编程建立 3 个并发协作进程或线程，它们分别完成 f(x,y)、f(x)、f(y)  实验代码：  #include **<stdio.h>** #include **<unistd.h>** #include **<stdlib.h>** #include **<pthread.h>** #include**<iostream> using namespace** std;  **int** pipFxin[2]; **int** pipFxout[2]; **int** pipFyin[2]; **int** pipFyout[2]; **int** pipFxyin[2]; **int** pipFxyout[2];  **void** \*taskFx(**void** \*arg) {  **while** (**true**) {  **int** x, ans = 1;  read(pipFxin[0], &x, **sizeof**(**int**));  **if** (x == 0)**break**;  **for** (**int** a = 1; a <= x; ++a) {  ans \*= a;  }  write(pipFxout[1], &ans, **sizeof**(**int**));  }  close(pipFxin[0]);  close(pipFxout[1]);  }  **void** \*taskFy(**void** \*arg) {  **while** (**true**) {  **int** x;  read(pipFyin[0], &x, **sizeof**(**int**));  **if** (x == 0)**break**;  **int** \*array = **new int**[x + 2];  array[1] = 1;  array[2] = 1;  **for** (**int** a = 3; a <= x; ++a) {  array[a] = array[a - 1] + array[a - 2];  }  write(pipFyout[1], &array[x], **sizeof**(**int**));  **delete**[] array;  }  close(pipFyin[0]);  close(pipFyout[1]);   } **void** \*taskFxy(**void** \*arg) {   **while** (**true**) {  **int** xy[2];  **int** z;  cout<<**"wait read xy"**<<endl;  read(pipFxyin[0], xy, 2\***sizeof**(**int**));  cout<<**"readout x="**<<xy[0]<<**" y="**<<xy[1]<<endl;  **if** (xy[0] == 0 || xy[1]==0){  **int** zero=0;  write(pipFxin[1], &zero, **sizeof**(**int**));  write(pipFyin[1], &zero, **sizeof**(**int**));  **break**;  }  write(pipFxin[1], &xy[0], **sizeof**(**int**));  write(pipFyin[1], &xy[1], **sizeof**(**int**));  **int** x,y;  read(pipFxout[0], &x, **sizeof**(**int**));  read(pipFyout[0], &y, **sizeof**(**int**));    **int** ans=x+y;  write(pipFxyout[1], &ans, **sizeof**(**int**));  } *// close(pipFyout[0]); // close(pipFxyout[1]); // close(pipFxout[0]); // close(pipFyin[1]); // close(pipFxin[1]); // close(pipFxyin[0]); //* }  **int** main(**int** argc, **char** \*arg[]) {  **int** ret;    pipe(pipFxin);  pipe(pipFxout);  pipe(pipFyin);  pipe(pipFyout);  pipe(pipFxyin);  pipe(pipFxyout);  pthread\_t threads[3];  pthread\_create(&threads[0], **NULL**, taskFx, (**void** \*) &ret);  pthread\_create(&threads[1], **NULL**, taskFy, (**void** \*) &ret);  pthread\_create(&threads[2], **NULL**, taskFxy, (**void** \*) &ret);    cout<<**"main pipFxyin"**<<endl;  **int** xy[2];  xy[0]=3;  xy[1]=2;  write(pipFxyin[1], xy, 2\***sizeof**(**int**));  **int** ans;  read(pipFxyout[0], &ans, **sizeof**(**int**));      cout<<**"ans="**<<ans<<endl;  xy[0]=0;  write(pipFxyin[1], xy, 2\***sizeof**(**int**));     exit(**EXIT\_SUCCESS**); }  运行结果： | | | |
| 结论分析与体会：  通过6个管道控制主进程+3个子进程的通信  主进程⬄f(x,y)  f(x,y)⬄f(x)  f(x,y)⬄f(y)  通过主进程向f(x,y)进程中输入两个参数  f(x,y)调用两个子进程进行运算 返回结果后加和 | | | |