**Практическая работа**

**«Организация взаимодействия процессов с помощью каналов»**

**Цели и задачи**

Научиться использовать каналы для организации взаимодействия процессов и их синхронизации. Изучить переназначение операций ввода-вывода.

**Общие сведения**

**Полудуплексные неименованные каналы**

Канал - средство связи вывода одного процесса с вводом другого процесса. Таким образом, каналы предоставляют метод односторонних коммуникаций между процессами, отсюда термин - полудуплексные.

Когда процесс создает канал, ядро устанавливает два файловых дескриптора для пользования этим каналом. Первый дескриптор используется, чтобы открыть путь ввода в канал (запись), в то время как второй применяется для получения данных из канала (чтение).

Данные, идущие через канал, проходят через ядро. В операционной системе каналы представлены корректным inode - индексным дескриптором, который существует в пределах самого ядра, а не в какой-либо физической файловой системе.

**Функции для работы с неименованными каналами**

Для использования функций работы с каналами необходимо подключить заголовочный файл **unistd.h**.

**int pipe (int fd[2])**

Функция создает полудуплексный канал, и возвращает 0 в случае успеха или -1 в случае неудачи. В fd[0] записывается дескриптор для чтения из канала, в fd[1] записывается дескриптор для записи в канал.

**size\_t read (int fd, void \*buf, size\_t count)**

45

Функция считывает через файловый дескриптор fd данные в buf размерностью count байт. Функция возвращает количество считанных байт. Если доступны данные размерностью менее count, они будут считаны и функция завершится. В том случае, когда нет данных, функция будет ожидать их поступления.

**size\_t write (int fd, void \*buf, size\_t count)**

Функция записывает через файловый дескриптор fd данные buf размерностью count байт. Функция возвращает количество записанных байт.

**int close (int fd)**

Функция закрывает дескриптор fd.

**Пример работы с каналом**

Процесс - родитель создает канал и порождает дочерний процесс, который посылает свой идентификатор (pid) в канал. Родительский процесс считывает данные из канала и выводит их на экран.

#include <stdio.h>

#include <unistd.h>

int main()

{

char s[15] ; *//для передачи и получения данных*

int fd[2] ; *//для получения дескрипторов канала*

if (pipe(fd)<0) *//если не удалось создать канал*

{ fprintf(stdout,"\nОшибка создания канала") ; return 0 ; }

if (fork()==0)

{//программный код для дочернего процесса

int r = sprintf(s,"MyPid=%d",getpid()) ; *//записать свой pid в s, r–длина строки*

write(fd[1],&s,r) ; *//послать данные s в канал через дескриптор для записи*

return 1 ;

}

read(fd[0],&s,15) ; *//прочитать данные из канала через дескриптор для чтения*

fprintf(stdout,"\nParent read - '%s'", s) ; *//вывести полученную строку на экран*

close(fd[0]) ; close(fd[1]) ; *//закрыть дескрипторы канала*

return 1 ;

}

**Именованные каналы (FIFO - каналы)**

Именованный канал – это отдельный тип файла в файловой системе, используемый для однонаправленного обмена информацией между процессами. Неименованный канал может быть использован только родственными процессами, тогда как именованный канал доступен и для независимых процессов, при условии, что всем этим процессам известно расположение и имя файла именованного канала. В mc именованные каналы отображаются с префиксом |.

**Функции для работы с именованными каналами**

Для использования функций работы с именованными каналами необходимо подключить заголовочный файл **fcntl.h**.

**int mkfifo (char \*name, int mode)**

Функция создает именованный канал с именем name и возвращает 0 в случае успеха или -1 в случае неудачи. В mode указываются флаги свойств создаваемого канала.

**int open (char \*name, int flags)**

Функция возвращает дескриптор файла с именем name. При работе с каналом переменная flags может содержать следующие флаги.

O\_RDONLY – открыть файл только для чтения

O\_WRONLY – открыть файл только для записи

O\_NONBLOCK – запрещает блокировку процесса при работе с каналом, как для функций **read** и **write,** так и при получении дескриптора канала. Если этот флаг не указан, то процесс, получающий дескриптор для чтения(записи),

47

блокируется до того времени, когда другой процесс не запросит дескриптор для записи(чтения).

**int unlink (char \*name )**

Функция удаляет файл с именем name.

Чтение и запись данных в именованный канал происходит с помощью функций **read** и **write**, описанных выше.

**Примеры работы с именованным каналом**

Процесс - родитель создает именованный канал и порождает дочерний процесс, который посылает свой идентификатор (pid) в канал. Родительский процесс считывает данные из канала и выводит их на экран.

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/wait.h>

int main()

{

char s[15] ; *//для передачи и получения данных*

int fd[2] ; *//для получения дескрипторов канала*

if (mkfifo("mypipe",S\_IFIFO|0666)<0) *//если не удалось создать канал*

{ fprintf(stdout,"\nОшибка создания канала") ; return 0 ;}

*//получить дескриптор для чтения*

*//если не указать флаг O\_NONBLOCK, процесс заблокирует сам себя*

fd[0] = open("mypipe",O\_RDONLY|O\_NONBLOCK) ;

*//получить дескриптор для записи*

fd[1] = open("mypipe",O\_WRONLY) ;

if (fork()==0)

{//программный код для дочернего процесса

int r = sprintf(s,"MyPid=%d",getpid()) ; *//записать свой pid в s, r–длина строки*

write(fd[1],&s,r) ; *//послать данные s в канал через дескриптор для записи*

48

return 1 ;

}

wait(NULL) ; */\*ожидание дочернего процесса необходимо, так как функция чтения из канала стала не блокирующей, т.е. если дочерний процесс не успеет записать данные в канал, функция чтения не получит данных и завершиться\*/*

read(fd[0],&s,15) ; *//прочитать данные из канала через дескриптор для чтения*

fprintf(stdout,"\nParent read - '%s'", s) ; *//вывести полученную строку на экран*

close(fd[0]) ; close(fd[1]) ; *//закрыть дескрипторы канала*

unlink(“mypipe”) ; *//удалить канал*

return 1 ;

}

Первое приложение создает именованный канал, второе приложение (писатель) получает дескриптор канала на запись и записывает в него свой идентификатор (pid). Первое приложение (читатель) ожидает поступления данных в канал и выводит их на экран. Для совместной работы приложений, исходя из особенностей алгоритма, необходимо сначала запустить первое приложение.

Приложение – читатель.

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

int main()

{

char s[15] ; *//для получения данных*

int fd ; *//для получения дескриптора канала*

mkfifo("mypipe",S\_IFIFO|0666) ; *//создать канал*

fd = open("mypipe",O\_RDONLY) ; *//получить дескриптор для чтения*

read(fd,&s,15) ; *//прочитать данные из канала (здесь функция read - блокирующая)*

49

fprintf(stdout,"\nрПрочитано : '%s'",s) ; *//вывести полученные данные на экран*

close(fd) ; *//закрыть дескриптор*

unlink("mypipe") ; *//уничтожить канал*

return 1 ;

}

Приложение – писатель

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

int main()

{

char s[15] ; *//для записи данных*

int fd ; *//для получения дескриптора канала*

fd = open("mypipe",O\_WRONLY) ; *//получить дескриптор для записи*

sprintf(s,"MyPid=%d",getpid()) ; *//записать свой pid в s*

write(fd,&s,15) ; *//записать s в канал*

close(fd) ; *//закрыть дескриптор*

return 1 ;

}

**Переназначение операций ввода-вывода**

Каждый создаваемый процесс имеет доступ к стандартным портам ввода-вывода.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Ввод | Вывод | |
| FILE\* | stdin | stdout | stderr |
| дескриптор | 0 | 1 | 2 |
| связан с | клавиатурой | дисплеем | дисплеем |

Каждый стандартный порт может быть переназначен в файл или канал, посредством функций dup2.

**int dup2 (int fd, int fd2)**

Функция делает fd2 копией дескриптора fd.

Таким образом, можно переназначить стандартный вывод в канал, посредством выполнения функции dup2(fd[1], 1), где fd[1] – дескриптор для записи в канал.

Также можно переназначить стандартный ввод из канала, посредством выполнения функции dup2(fd[0], 0), где fd[0] – дескриптор для чтения из канала.

**Порядок выполнения практической работы**

1.Разработать алгоритм решения задания а, с учетом разделения вычислений между несколькими процессами. Для обмена информацией между процессами использовать неименованные каналы.

2.Реализовать алгоритм решения задания а и протестировать на нескольких примерах.

3.Разработать алгоритм решения задания б, разделив вычисления между несколькими приложениями. Для обмена информацией между приложениями использовать именованные каналы.

4.Реализовать алгоритм решения задания б и протестировать на нескольких примерах.

5.Написать приложение, переназначающее стандартный вывод в канал и стандартный ввод из канала, и замещаемое приложением задания а.

**Варианты заданий**

1.а) Определить является ли матрица А магическим квадратом. Входные данные: целое положительное число n, массив чисел А размерности nxn. Матрица является магическим квадратом, когда равны между собой суммы всех строк и суммы всех столбцов. Использовать n или n+1 процессов для решения задачи.

б) Заменить наиболее часто встречающийся символ в строке S1, наиболее часто встречающимся символом в строке S2 и наоборот. Входные данные первого приложения: строка символов S1 произвольной длины. Входные данные второго приложения: строка символов S2 произвольной длины.

2.а) В строке символов S заменить каждый ai-й символ на bi символ. Входные данные: целое положительное число n, пары символов (a1, b1), (a2, b2), … (an, bn), строка символов S произвольной длины. Использовать n или n+1 процессов для решения задачи.

б) Вычислить скалярное произведение вектора А на вектор B. Входные данные первого приложения: массив чисел А фиксированной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

3. а) Вычислить произведение матриц А и B. Входные данные: целое положительное число n, массивы чисел А и B размерности nxn. Использовать n или n+1 процессов для решения задачи.

б) Первое и второе приложение ждут ввода числа от 1 до 10 пользователем. После чего обмениваются полученными значениями. Первое приложение подсчитывает количество совпавших значений, второе приложение находит произведение полученной пары чисел. Ввод чисел пользователями должен быть согласован.

4. а) Вычислить векторное произведение вектора А на вектор B. Входные данные: массивы чисел А и В размерности 3. Использовать не менее трех процессов для решения задачи.

б) Первое приложение заменяет символ а в строке S на символ b. Второе приложение ожидает ввода символа а, третье приложение ожидает ввода символа b. Входные данные первого приложения: строка S произвольной длины. Входные данные второго приложения: символ а. Входные данные третьего приложения: символ b. Время работы приложений не ограничено.

5. а) Определить совпадает ли хотя бы одна пара – сумма i-й строки и i-го столбца матрицы А. Входные данные: целое положительное число n, массив чисел А размерности nxn. Использовать n или n+1 процессов для решения задачи.

б) Первое приложение находит сумму элементов массива А и В, второе приложение находит произведение элементов массива А и В. Входные данные первого приложения: массив чисел А фиксированной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

6. а) Определить совпадает ли хотя бы одна пара – сумма i-й строки и i-го столбца матрицы А. Входные данные: целое положительное число n, массив чисел А размерности nxn. Использовать два процесса для решения задачи.

б) Первое приложение случайным образом определяет число А. Второе приложение ожидает ввода числа В пользователем, третье приложение ожидает ввода числа С. Второе и третье приложения посылают числа первому, оно определяет, совпадает ли полученное число с числом А, и отсылает назад ответ. Если ответ положительный, пользователю присуждается очко. Игра ведется 15 раундов. Каждый пользователь должен узнавать об успехах другого на каждом раунде. Работа приложений должна быть согласована.

7. а) Вычислить скалярное произведение вектора А на вектор B. Входные данные: целое положительное число n, массивы чисел А и В размерности n, целое положительное число к от 1 до n/2. Использовать n/к процессов для решения задачи.

б) Заменить наиболее часто встречающийся символ в строке S, символом а1. Затем заменить наиболее часто встречающийся символ в строке S, отличный от а1, символом а2. Продолжать до тех пор, пока работает второе приложение. Входные данные первого приложения: строка символов S произвольной длины. Входные данные второго приложения: символы ai. Второе приложение, получив ai символ, выводит получившуюся строку S на экран. Количество символов, которое можно ввести ограничено длиной строки S. Работа приложений должна быть согласована.

8. а) Найти максимальный элемент в матрице А. Входные данные: целые положительные числа n и к, массив чисел А размерности nxк.. Использовать n или к процессов для решения задачи.

б) Определить является ли каждый i-й элемент массива А больше i-го элемента массива В. Входные данные первого приложения: массив чисел А фиксированной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

9. а) Проложена дорога ведущая от города А к городу Б, от Б к В, от В к Г, от Г к Д. В каждом городе есть некоторое число пассажиров, желающих поехать в другие города. Но автобусы ходят только между соседними городами, каждый автобус вмещает в себя только 20 пассажиров. Необходимо доставить всех пассажиров в пункты назначения, отображая при этом происходящие изменения. Использовать 4 или 5 процессов для решения задачи.

б) Определить все символы, содержащиеся как в строке S1, так и в строке S2. Входные данные первого приложения: строка символов S1 произвольной длины. Входные данные второго приложения: строка символов S2 произвольной длины.

10. а) Найти индексы i и j, для которых существует наибольшая последовательность a[i] – a[i+1] + a[i+2] – a[i+3] … +/- a[j]. Входные данные: целое положительное число n, массив чисел А размерности n.

б) Первое приложение ожидает ввода чисел a, b, c и отсылает их второму приложению, которое находит решение уравнения ax2+bx+c=0, и отсылает результат первому приложению.

11. а) Найти столбец и строку с минимальными суммами в матрице А. Входные данные: целые положительные числа n и к, массив чисел А размерности nxк.. Использовать n или к процессов для решения задачи.

б) В первом и втором приложении пользователи вводят массивы чисел 5х5, содержащие нули и единицы, количество единиц должно быть не менее десяти. На каждом ходе пользователи делают предположение о расположении в массиве другого пользователя единицы – вводят номер строки и номер столбца. Побеждает тот, кто первым найдет единицу в массиве соперника. Работа приложений должна быть согласована.

12. а) Определить какая сумма элементов массива А является максимальной, сумма всех простых чисел или сумма всех четных чисел. Входные данные: целое положительное число n, массив чисел А размерности n. Использовать два процесса для решения задачи.

б) Два приложения обмениваются случайными тройками чисел (a, b, c), до тех пор, пока две троки не окажутся равными между собой без учета порядка. Работа приложений должна быть согласована.

13. а) В двоичном массиве А определить индексы х1, у1, х2, у2 с максимальным значением (х2-х1)+(у2-у1) для которых существует множество одинаковых между собой элементов, заключенных в «прямоугольнике» с верхним левым углом (х1, у1) и нижним правым углом (х2, у2).

б) Определить все индексы i для которых элемент массива А равен элементу массива В. Входные данные первого приложения: массив чисел А фиксированной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

14. а) Определить, равны ли между собой суммы двух главных диагоналей в матрице А. Входные данные: целое положительное число n, массив чисел А размерности n. Использовать два процесса для решения задачи.

б) Первое приложение находит все символы, содержащиеся в строке S1, и не содержащиеся в строке S2. Второе приложение находит все символы, содержащиеся в строке S2, и не содержащиеся в строке S1. Входные данные первого приложения: строка символов S1 произвольной длины. Входные данные второго приложения: строка символов S2 произвольной длины.

15. а) Задана строка S, содержащая не менее двух слов. Необходимо найти слово, содержащее максимальное количество вхождений символа а. Входные данные: строка S, символ а. Для решения задачи использовать столько процессов, сколько слов в строке.

б) Поменять местами соответственные элементы в массивах А и В, если хотя бы один элемент является простым числом. Входные данные первого приложения: массив чисел А фиксированной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

16. а) В матрице А найти строку с максимальным произведением. Входные данные: целое положительное число n, массив чисел А nxn. Использовать n или n+1 процессов для решения задачи.

б) Первое приложение обладает банком вопросов и ответов (не менее 10). Второе и третье приложение запускаются игроками. Первое приложение отправляет один и тот же случайный вопрос обоим игрокам и ожидает ответы. Игрок, ответивший правильно, получает очко. Игра ведется в три хода. В завершении игроки узнают результат игры.

17. а) Найти максимальное простое число в массиве А. Входные данные: целое положительное число n > 4, массив чисел А размерности n. Использовать четыре процесса для решения задачи.

б) Первое приложение содержит список студентов по предмету – фамилия, оценки. Пользователь, запустив второе приложение, может послать запрос а) вывести фамилию самого успешного студента (по среднему баллу) б) вывести фамилию самого неуспешного студента (по среднему баллу) в) добавить указанную оценку указанному студенту.

18. а) Найти наиболее часто встречающуюся сумму строки матрицы А. Входные данные: целое положительное число n, массив чисел А размерности nхn. Использовать n или n+1 процессов для решения задачи.

б) Первое приложение заменяет максимальный элемент в массиве А, минимальным элементом из массива В. Второе приложение заменяет максимальный элемент в массиве В, средним арифметическим элементов массива А. Входные данные первого приложения: массив чисел А произвольной размерности. Входные данные второго приложения: массив чисел B, той же размерности, что и массив А.

19. а) Вычислить суммы элементов главной диагонали, элементов, стоящих ниже главной диагонали и элементов, стоящих выше главной диагонали. Определить минимальную и максимальную сумму. Входные данные: целое положительное число n>2, массив чисел А размерности nхn. Использовать 3 или 4 процесса для решения задачи.

б) Составить строку S3, добавив в нее элементы S1[i] и S2[i], отличные между собой. Входные данные первого приложения: строка символов S1 произвольной длины. Входные данные второго приложения: строка символов S2 произвольной длины.

20. а) Задана строка S, содержащая не менее двух чисел. Необходимо найти наибольшее и наименьшее число. Входные данные: строка S произвольной длины. Для решения задачи использовать столько процессов, сколько чисел записано в строке.

б) В первом и втором приложении игроки бросают два кубика. Очко зачисляется тому игроку, для которого сумма выпавших значений больше или же игроку, с одинаковыми выпавшими значениями. Если оба игрока имеют одинаковые выпавшие им значения, побеждает тот, чья сумма больше. Игра ведется в 8 ходов. На каждом ходе каждый игрок должен получать информацию о своих очках и очках противника. Работа приложений должна быть согласована.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |