Лабораторная работа 7

Семафоры

1. Семафоры

Для синхронизации работы процессов и для синхронизации доступа нескольких процессов к общим ресурсам используются семафоры. Общими ресурсами процессов являются файлы, сегменты разделяемой памяти. Возможность одновременного изменения несколькими процессами общих данных называют критической секцией, так как такая совместная работа процессов может привести к возникновению ошибок. Например, если несколько процессов осуществляют запись данных в один и тот же файл, эти данные могут оказаться перемешанными. Наиболее простой механизм защиты критической секции состоит в расстановке «замков», пропускающих только один процесс для выполнения критической секции, и останавливающий все остальные процессы, пытающиеся выполнить критическую секцию, до тех пор, пока эту критическую секцию не выполнит пропущенный процесс. Семафоры позволяют выполнять такую операцию, как и многие другие.

Под семафором может пониматься как единичный семафор, так и несколько семафоров, объединенных в группу. Общий механизм действия семафора таков:

семафор обладает внутренним значением – числом, из множества целых чисел с нижней границей (например – с нулем), процессы могут изменять значение семафора – увеличивать или уменьшать. Если процесс изменяет значение семафора и выходит за граничное значение, такой процесс приостанавливается, пока какой-нибудь другой процесс не изменит значение семафора так, чтобы заблокированный процесс смог выполнить изменение значения семафора.

Использование общих данных несколькими процессами может привезти к ошибкам и конфликтам. Но при этом семафоры и сами являются общими данными. Такое положение не является противоречивым, в силу того, что:

- значение семафора расположено не в адресном пространстве некоторого процесса, а в адресном пространстве ядра;

- операция проверки и изменения значения семафора, вызываемая процессом является атомарной, т.е. непрерываемой другими процессами, эта операция выполняется в режиме ядра.

Общими данными процессов также являются каналы и сообщения. Но операции с каналами и сообщениями защищаются системой, и, как правило, программист не должен использовать семафор для защиты записи сообщения в очередь сообщений, либо записи данных в канал. Однако это не всегда правильно. Например, система обеспечивает атомарную запись в канал, только для данных не больше определенного объема.

Семафоры используются и для синхронизации потоков. В многопоточном приложении критической секцией является возможность изменения глобальной переменной несколькими потоками.

2. Функции для работы с семафорами

Семафор обладает внутренним значением - целым числом, принадлежащим типу unsigned short. Семафоры могут быть объединены в единую группу.

Рассмотрим функции для работы с семафорами (заголовочные файлы – *sys/ipc.h* и *sys/sem.h*).

int semget (key_t key, int nsems, int semflag)

Функция создает группу семафоров с ключом key (или дает доступ к уже существующей группе с заданным ключом), состоящую из nsems семафоров. Параметр key может быть равен IPC_PRIVATE, в этом случае, система сама определяет незанятый ключ для группы семафоров. Параметр semflag может быть равен IPC_CREAT – создать группу семафоров, или IPC_EXCL – получить доступ к существующей группе. Если же shmflag равен IPC_CREAT | IPC_EXCL, функция создаст новую группу семафоров с ключом key, только когда не существует другой группы семафоров с тем же ключом. Параметр shmflag также может включать флаги доступа. Функция возвращает дескриптор группы семафоров в случае успеха, или -1 в случае неудачи. Все семафоры в созданной группе имеют внутреннее значение ноль.

*int semop(int semid, sembuf *semop, size_t nops)*

Функция выполняет над группой семафоров с дескриптором semid набор операций semop, nops – количество операций, выполняемых из набора semop.

Для задания операции над группой семафоров используется структура sembuf.

Первый параметр структуры sembuf определяет порядковый номер семафора в группе. Семафоры в группе индексируются с нуля.

Второй параметр структуры sembuf представляет собой целое число = S, и определяет действие, которое необходимо произвести над семафором, с индексом, записанным в первом параметре.

Если S>0 к внутреннему значению семафора добавляется число S. Эта операция не блокирует процесс.

Если S=0 процесс приостанавливается, пока внутреннее значение семафора не станет равно нулю.

Если S<0 процесс должен отнять от внутреннего значения семафора модуль S. Если значение семафора - $|S| \geq 0$ производится вычитание и процесс продолжает свою работу. Если значение семафора - $|S| < 0$ процесс останавливается до тех пор, пока другой процесс не увеличит значение семафора на достаточную величину, чтобы операция вычитания выдала неотрицательный результат. Тогда производиться операция вычитания и процесс продолжает свою работу. Например, если значение семафора равно трем, а процесс пытает выполнить над ним операцию -4, этот процесс будет заблокированным, пока значение семафора не увеличиться хотя бы на единицу.

Третий параметр структуры sembuf может быть равен 0 – тогда операция S≤0 будет предполагать блокировку процесса, т.е. выполняться так, как описано выше. Также sembuf может быть равен IPC_NOWAIT, в этом случае, работа процесса не будет останавливаться. Если процесс будет пытаться выполнить вычитание от значения семафора дающее отрицательный результат, эта операция просто игнорируется и процесс продолжает выполнение.

3. Примеры использования семафора

Пример использования семафора для синхронизации процессов

Процесс-родитель создает четыре процесса-потомка и ожидает их завершения, используя для этого семафор.

```
#include <stdio.h>

#include <unistd.h>

#include <sys/sem.h>

#include <sys/ipc.h>

int main()

{       int semid ; //для хранения дескриптора группы семафоров

        //создать группу семафоров, состоящую из одного семафора

        semid = semget (IPC_PRIVATE, 1, IPC_CREAT|0666) ;

        if ( semid < 0 ) //если не удалось создать группу семафоров, завершить выполнение

        { fprintf(stdout, “\nОшибка”) ; return 0 ; }

        sembuf  Plus1 = {0, 1, 0} ; //операция прибавляет единицу к семафору с индексом 0

sembuf  Minus4 = {0, -4, 0} ; //операция вычитает 4 от семафора с индексом 0

//создать четыре процесса-потомка
```

```
for (int i=0 ; i<4 ; i++)
{       if  ( fork() == 0 ) //истинно для дочернего процесса
        {  //здесь должен быть код, выполняемый процессом-потомком
           //добавить к семафору единицу, по окончанию работы
           semop( semid, &Plus1, 1) ; return 1 ;
        }
}
semop( semid, &Minus4, 1) ; return 1 ;
}
```

В описанном примере, созданный семафор при создании обладает нулевым значением. Каждый из четырех порожденный процессов, после выполнения своих вычислений, увеличивает значение семафора на единицу. Родительский процесс пытается уменьшить значение семафора на четыре. Таким образом, процесс-родитель останется заблокированным, до тех пор, пока не отработают все его потомки.

Последний параметр в функции semop определяет количество операций, берущихся для выполнения из второго параметра функции. Т.е. следующий вызов

```
sembuf  Minus4 = {0, -4, 0} ;
semop( semid, &Minus4, 1) ;
```

Нельзя заменить таким вызовом.

```
sembuf  Minus1 = {0, -1, 0} ;
semop( semid, &Minus1, 4) ;
```

Корректной заменой может являться.

```
sembuf  Minus1[4] = {0, -1, 0,      0, -1, 0,      0, -1, 0,      0, -1, 0} ;
semop( semid, Minus1, 4) ;
```

Пример использования семафора для защиты критической секции

Воспользуемся семафором для синхронизации доступа нескольких процессов к общему ресурсу, т.е. для защиты критической секции. Общим ресурсом будет являться разделяемая память.

Процесс – родитель создает сегмент разделяемой памяти и порождает три дочерних процесса, процесс-родитель и его потомки вычисляют сумму элементов определенной части массива и записывает вычисленную сумму в разделяемую память. Родительский процесс дожидается окончания работы потомков, и выводит окончательный результат – сумму всех элементов массива.

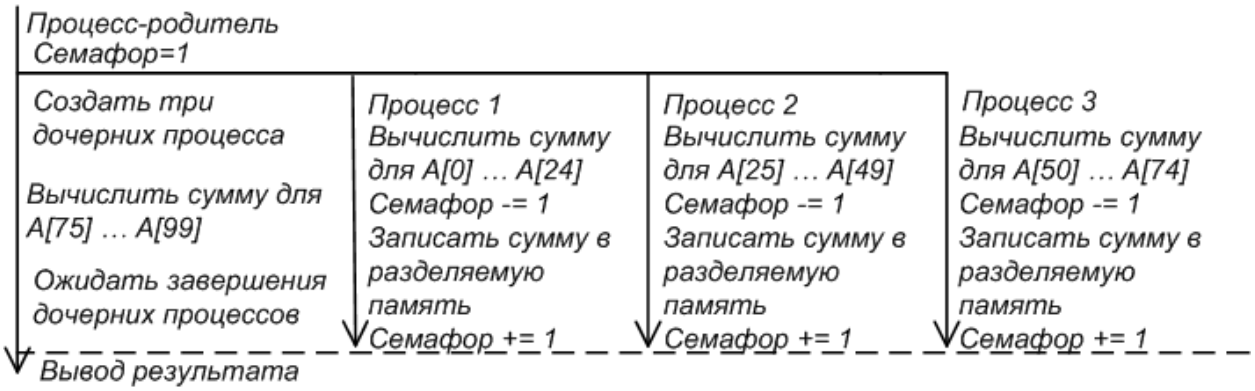

Рисунок - Схема процессов

Так как четыре процесса могут изменять данные разделяемой памяти, необходимо сделать это изменение атомарным для каждого процесса. Для этого создадим семафор, который будет принимать два значения – ноль и единицу.

```
#include <stdio.h>

#include <unistd.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <sys/sem.h>

#include <sys/wait.h>

int shmid ; //для хранения дескриптора разделяемой памяти

int semid ; //для хранения дескриптора группы семафоров

sembuf  Plus1 = {0, 1, 0} ; //операция прибавляет единицу к семафору с индексом 0
```

```
sembuf  Minus1 = {0, -1, 0} ; //операция вычитает единицу  от семафора с индексом 0

int A[100] ; //массив, сумма элементов которого вычисляется процессами

struct mymem   //структура, под которую будет выделена разделяемая память

{   int sum ; //для записи суммы

} *mem_sum ;

void summa (int p) //для вычисления суммы части элементов массива

{          int i, sum = 0 ; //для суммирования элементов

int begin =25*p ; //индекс массива, с которого начинается суммирование

int end = begin+25 ; //индекс массива, на котором завершается суммирование

for(i=begin ; i<end ; i++) sum+=A[i] ; //вычисление суммы части элементов массива

semop( semid, &Minus1, 1) ; //отнять единицу от семафора

mem_sum->sum+=sum; //добавление вычисленного результата в общую переменную

          semop( semid, &Plus1, 1) ; //добавить единицу к семафору

}

int main()

{          //запрос на создание разделяемой памяти объемом 2 байта

shmid = shmget(IPC_PRIVATE, 2, IPC_CREAT|0666) ;

//если запрос оказался неудачным, завершить выполнение

if (shmid < 0 ) { fprintf(stdout,"\nОшибка") ; return 0 ; }

//создать группу семафоров, состоящую из одного семафора

semid = semget (IPC_PRIVATE, 1, IPC_CREAT|0666) ;

//если не удалось создать группу семафоров, завершить выполнение

          if ( semid < 0 ) { fprintf(stdout, “\nОшибка”) ; return 0 ; }
```

```
semop( semid, &Plus1, 1) ; //теперь семафор равен единице

//теперь mem_sum указывает на выделенную разделяемую память

mem_sum = (mymem *)shmat(shmid,NULL,0) ;

mem_sum->sum = 0 ;

        //тут должна быть инициализация элементов массива A

//создать три  процесса-потомка

for (int i=0 ; i<3 ; i++)

{       if  ( fork() == 0 ) //истинно для дочернего процесса

        {  summa(i) ; return 1 ; }

}

summa(3) ; //родительский процесс вычисляет последнюю четверть массива

        for (int i=0 ; i<3 ; i++)

wait(NULL) ; //дождаться завершения процессов-потомков

//вывести на экран сумму всех элементов массива

fprintf(stdout,"\nРезультат =  %d",mem_sum->sum) ;

return 1;

}
```

Семафор получает при создании значение равное нулю, которое сразу же устанавливается в единицу. Первый процесс, вызвавший функцию semop(semid, &Minus1, 1), уменьшает значение семафора до нуля, переходит к записи в разделяемую память, и по завершении операции записи, устанавливает значение семафора в единицу, вызвав semop(semid, &Plus1, 1). Если управление перейдет к другому процессу, во время записи в разделяемую память первым процессом, вызов функции «отнять от семафора единицу» остановит работу другого процесса, до того момента, когда значение семафора не станет положительным. Что может произойти только тогда, когда первый процесс завершит запись в общую память, и выполнит операцию – добавить к семафору единицу.

Если процессы обладают несколькими общими ресурсами, то необходимо для каждого общего ресурса создавать свой семафор.

4. Цели и задачи

Познакомиться с общими принципами работы семафоров. Научиться использовать семафоры для синхронизации процессов и потоков, и для защиты критических секций.

5. Порядок выполнения лабораторной работы

1. Разработать алгоритм решения задания, с учетом разделения вычислений между несколькими процессами. Для обмена информацией между процессами использовать разделяемую память. Для защиты операций с разделяемой памятью и синхронизации процессов использовать семафоры. Реализовать алгоритм и протестировать его на нескольких примерах.

2. Посмотреть в динамике работу семафоров для созданных приложений, используя команду ipcs -s.

6. Варианты заданий (1-15)

1.

Задана строка S, содержащая не менее двух слов. Необходимо найти среди слов, палиндром максимальной длины. Входные данные: строка S. Для решения задачи использовать столько процессов, сколько слов в строке. Палиндромом является фраза или слово, одинаково читаемая как слева направо, так и справа налево, пример – поп.

2.

Найти максимальный M и минимальный элемент m массива A и составить множество чисел, лежащих в интервале (m, M) и не содержащихся в массиве A. Входные данные: целое положительное число n, массив чисел A размерности n.

3.

Определить какая сумма больше, сумма всех положительных чисел, стоящих выше главной диагонали и ниже второй главной диагонали, или сумма модулей всех отрицательных чисел, стоящих выше второй главной диагонали и ниже главной диагонали. Входные данные: целое положительное число n, массив чисел A размерности nxn.

4.

Найти среднее арифметическое всех «особых» элементов матрицы A. Будем считать, элемент особым, если он больше суммы всех остальных элементов, стоящих в том же столбце. Входные данные: целое положительное число n, массив чисел A размерности nxn. Использовать n или n+1 процессов для решения задачи.

5.

Найти сумму индекса строки с максимальным элементом на главной диагонали с индексом столбца с минимальным элементом на второй главной диагонали. Входные данные: целое положительное число n, массив чисел A размерности nxn.

6.

Составит строку из максимальных и минимальных элементов строк матрицы A. Порядок элементов в строке не важен. Входные данные: целое положительное число n, целое положительное число к, массив чисел от 0 до 9 A размерности nxк. Использовать n или n+1 процессов для решения задачи.

7.

Определить количество чисел m, являющихся квадратами некоторого целого числа, в матрице A. Заменить все простые числа в A на m. Входные данные: целое положительное число n, массив чисел A размерности nxn.

8.

Найти сумму индексов всех седловых точек матрицы A. Будем считать, элемент седловой точкой, если он является наименьшим в своей строке и наибольшим в своем столбце, либо наоборот. Входные данные: целое положительное число n, массив чисел A размерности nxn.

9.

Найти максимальный и минимальный среди тех элементов матрицы A, сумма индексов которых равна двойке в любой целочисленной степени. Входные данные: целое положительное число n, целое положительное число к, массив чисел от A размерности nxк.

10.

Определить является ли матрица A симметричной относительно главной диагонали. Входные данные: целое положительное число n, массив чисел A размерности nxn. Использовать не менее 4 процессов для решения задачи.

11.

В матрице A найти в каждой строке наибольший элемент и поменять его местами с элементом, стоящим на главной диагонали и в той же строке. Входные данные: целое положительное число n, массив чисел A размерности nxn. Использовать n или n+1 процессов для решения задачи.

12.

Упорядочить по возрастанию элементы в каждой строке матрицы A. Входные данные: целое положительное число n, целое положительное число к, массив чисел от A размерности nxк. Использовать n или n+1 процессов для решения задачи.

13.

В массиве строк хранятся фамилии и оценки учащихся. Найти учащегося с максимальным средним баллом, вывести список всех учащихся, имеющих однофамильцев. Входные данные: целое положительное число n, массив строк размерности n. Использовать n или n+1 процессов для решения задачи.

14.

Найти произведение всех элементов в матрице A, сумма или разность индексов которых является простым числом, отрицательные разности не рассматривать. Входные данные: целое положительное число n, целое положительное число к, массив чисел от A размерности nxк.

15.

Вычислить произведение матрицы A на B, где матрица B получена из матрицы A, заменой отрицательных элементов нулем и последующим транспонированием. Входные данные: целое положительное число n, массив чисел A размерности nxn.

7. Варианты заданий (16-30)

16.

Двоичные числа записаны в строке, разделителем является пробел. Количество чисел равно m. Найти сумму всех двоичных чисел как двоичное число и как десятичное число. Входные данные: строка S. Для решения задачи использовать не менее m процессов.

17.

Проверить, можно ли составить слово S из элементов символьного массива C. Учитывать количество требуемых символов для составления слова. Входные данные: строка S, массив символов C. Использовать для решения задачи столько процессов, сколько неповторяющихся символов в строке S.

18.

Строка содержит произвольный русский текст. Вывести сколько раз в ней встречается буква а, буква б, буква в и т.д. Найти три наиболее часто встречающиеся буквы. Входные данные: строка S. Использовать для решения задачи столько процессов, сколько букв в русском алфавите.

19.

Найти максимальный по модулю элемент в матрице A и его индексы – x, y. Поменять местами строку x со строкой k и столбец y со столбцом k. Входные данные: целое положительное число n, массив чисел A размерности nxn, целое положительное число k ≥ 0 и ≤ n-1.

20.

В массиве A заменить отрицательные элементы нулями, а положительные элементы единицами. Перевести полученное двоичное число в десятичную систему счисления. Входные данные: целое положительное число n, массив чисел A размерности n, целое положительное число к ≥ 2 и ≤ n/2. Использовать для решения задачи к процессов.

21.

Определить является ли матрица A магическим квадратом. Входные данные: целое положительное число n, массив чисел A 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.

22.

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.

23.

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

24.

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

25.

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

26.

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

27.

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

28.

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

29.

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.

30.

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 A размерности n.

8. Варианты заданий (31-45)

31.

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

32.

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

33.

В двоичном массиве A определить индексы x1, y1, x2, y2 с максимальным значением (x2-x1)+(y2-y1) для которых существует множество одинаковых между собой элементов, заключенных в «прямоугольнике» с верхним левым углом (x1, y1) и нижним правым углом (x2, y2).

34.

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

35.

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

36.

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

37.

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

38.

Найти наиболее часто встречающуюся сумму строки матрицы A. Входные данные: целое положительное число n, массив чисел A размерности nxn. Использовать n или n+1 процессов для решения задачи.

39.

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 A размерности nxn. Использовать 3 или 4 процесса для решения задачи.

40.

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.

41.

Задача разделения множеств. Заданы два множества натуральных чисел S и T. Сохраняя мощность множеств S и T, необходимо собрать в S наименьшие элементы множества $S \cup T$, а в T - наибольшие. Задача решается таким образом. Найдем максимальный элемент в множестве S и минимальный элемент в множестве T. Если максимум в множестве S больше минимума в множестве T, то максимальный элемент множества S исключается из S и добавляется в множество T, а минимальный элемент множества T исключается из T и включается в множество S. Иначе вычисления завершаются.

42.

Задана матрица чисел A размерности mxn и точность e. Итерация алгоритма состоит в вычислении всех не граничных элементов новой матрицы по следующей формуле $A'_{i,j} = (A_{i+1,j} + A_{i-1,j} + A_{i,j+1} + A_{i,j-1})/4$. Итерация алгоритма будет последней, если для любой пары не граничных элементов из вычисленной матрицы X и У, выполняется $|X-У|<e$.

43.

Задача о новобранцах. Строю новобранцев дается команда «налево» или «направо». Все новобранцы стараются исполнить приказ, но проблема в том, что они не знают где право, а где лево. Следовательно, каждый новобранец поворачивается либо направо, либо налево. Если новобранец повернулся и видит, что его сосед стоит к нему спиной, он считает, что все сделал правильно. Если же они сталкиваются лицом к лицу, то оба считают, что ошиблись, и разворачиваются на 180 градусов. Создать многопроцессное приложение, моделирующее поведение строя новобранцев, пока он не придет к стационарному состоянию. Количество новобранцев ≥ 100. Отдельный поток отвечает за часть строя не менее 50 новобранцев.

44.

Создать приложение с тремя процессами. Каждый процесс работает со своим массивом и сверяет сумму элементов массива с суммами элементов массивов других процессов, процессы останавливаются, когда все три суммы равны между собой. Если суммы не равны, каждый процесс прибавляет единицу к одному элементу массива или отнимает единицу от одного элемента массива, затем снова проверяет условие равенства сумм. На момент остановки всех трех процессов, суммы элементов массивов должны быть одинаковы.

45.

Задано произвольное число точек (x, y). Найти пару не одинаковых точек, расстояние между которыми минимально.

9. Варианты заданий (46-60)

46.

Найти суммы элементов квадратной матрицы, расположенных в «линиях», параллельных главной диагонали. Выделить максимальную сумму. Входные данные: число n, матрица A nxn.

47

Задано произвольное число окружностей (x, y, r). Найти все пары пересекающихся окружностей.

48.

Определить индексы особых элементов матрицы, элемент считается особым, если он равен хотя бы одному соседнему элементу из 8 его окружающих. Входные данные: числа n и к, матрица A nxк. Элементы, стоящие на границах матрицы, также требуется проверять.

49.

Задано произвольное количество слов и символов, определить какие из заданных слов, можно составить из указанных символов.

50.

Первый и второй процесс задают случайное число, если числа совпадают, работа процессов завершается, иначе процессы снова задают случайное число и т.д.

51.

Найти сумму индексов тех строк матрицы, сумма элементов которых больше К. Входные данные: числа n, к, К, матрица A nxк.

52.

Найти сумму тех трехзначных чисел, хотя бы две цифры которых меньше К. Входные данные: число К.

53.

Задано произвольное количество чисел, найти число с максимальной суммой образующих его цифр.

54.

Вычислить сумму всех четных столбцов матрицы, сумму всех нечетных строк матрицы, найти квадрат их разности. Входные данные: числа n, к, матрица A nxк.

55.

Разделить матрицу на четыре равные части, найти суммы их элементов, определить часть с максимальной суммой. Входные данные: число n, матрица A nxn.

56.

В ящике лежат 6 черных и 2 белых шара. На первом ходу игры первый и второй процесс достают по шару, если у обоих оказались белые шары, результатом игры считается ничья и игра завершается, если только у одного шар белый, такой процесс объявляется победителем и игра завершается, если у обоих шары черные начинается следующий аналогичный ход игры.

57.

Задано произвольное число точек (x, y) и число К. Найти пару с максимальной координатой x, такую что, расстояние от начала координат до этой точки меньше либо равно К.

58.

Из входной матрицы сформировать новую матрицу, каждый элемент которой равен среднему арифметическому 8и соседних элементов, элемента входной матрицы с аналогичными индексами. Учесть ситуацию граничных элементов. Входные данные: числа n и к, матрица A nxк. Выходные данные: матрица B nxк.

59.

Найти минимальный элемент среди особых элементов матрицы, элемент считается особым, если среди 8и его окружающих элементов, хотя бы три являются его делителями. Входные данные: числа n и к, матрица A nхк. Элементы, стоящие на границах матрицы, также требуется проверять.

60.

Задано произвольное число прямоугольников (a, b), записанных в файле, найти прямоугольник с максимальной площадью.