Управление процессами, потоками (нитями)

Лаб 1
а) Тема: Процессы, потоки (нити)
Задание: Составить программу (процесс), который параллельно запускает два потока (нити)
Назначение программы:
При запуске программы кнопкой "Пуск", одновременно параллельно с задачей запускаются два потока Tthread1 и TThread2, которые пытаются установить "бегунок" в свою позицию (1-в позицию 10, 2-й в позицию 90)
У потоков можно менять приоритеты, и в зависимости от приоритета предпочтение отдается либо одному, либо другому потоку
При завершение программы потоки уничтожаются
б) Тема: Управление процессами, потоками (нитями) в критической секции с помощью блокирующей переменной (простейшего семафора)
В задании использовать программу задания 1
В программе сделать следующие изменения:
Ввести глобальную переменную для семафора типа integer
Разместить кнопки ПУСК 1 и ПУСК 2 для запуска первого и второго потоков (нитей), перед этим устанавливается семафор в положение "занято"
Разместить кнопки СТОП 1 и СТОП 2 для остановки первого и второго потоков (нитей), устанавливается семафор в положение "свободно"
Кнопка ПУСК 1 задает для первого потока самый низкий приоритет
Кнопка ПУСК 2 задает для второго потока самый высокий приоритет
Описание работы программы
Потоки запускаются последовательно. Если работает один из потоков, то второй невозможно запустить, т.к. критическая секция занята. Сообщение "Занято потоком"
Кнопка СТОП освобождает критическую секцию и уничтожает текущий поток.
Кнопка ПУСК запускает поток и блокирует нажатие кнопки СТОП другого потока
Правильная работа программы заключается в следующем: Кнопка ПУСК 1 устанавливает бегунок в положение 10, там он и остается пока не нажимаем кнопку ПУСК 2, которая устанавливает его в положение 90
Обратить внимание! Семафор (блокирующая переменная) - глобальная переменная, доступная обоим потокам, следовательно они работают в одном адресном пространстве (данной задачи). Если бы семафор регулировал взаимодействие не потоков, а процессов, то он должен быть глобальным по отношению к ним и таким образом находиться в адресном пространстве операционной системы, которая и управляет процессами

Лаб 2
Тема: многопоточность.
а)**Первая задача о Винни-Пухе, или неправильные пчелы.** Неправильные пчелы, подсчитав в конце месяца убытки от наличия в лесу Винни-Пуха, решили разыскать его и наказать в назидание всем другим любителям сладкого. Для поисков медведя они поделили лес на участки, каждый из которых прочесывает одна стая неправильных пчел. В случае нахождения медведя на своем участке стая проводит показательное наказание и возвращается в улей. Если участок прочесан, а Винни-Пух на нем не обнаружен, стая также возвращается в улей. Требуется создать многопоточное приложение, моделирующее действия пчел. При решении использовать парадигму портфеля задач.
б) **Первая военная задача.** Темной-темной ночью прапорщики Иванов, Петров и Нечепорчук занимаются хищением военного имущества со склада родной военной части. Будучи умными людьми и отличниками боевой и строевой подготовки, прапорщики ввели разделение труда: Иванов выносит имущество со склада, Петров грузит его в грузовик, а Нечепорчук подсчитывает рыночную стоимость добычи. Требуется составить многопоточное приложение,

моделирующее деятельность прапорщиков. При решении использовать парадигму «производитель-потребитель» с активным ожиданием.

с) **Задача о Пути Кулака.** На седых склонах Гималаев стоят два древних буддистских монастыря: Гуань-Инь и Гуань-Янь. Каждый год в день сошествия на землю боддисатвы Араватти монахи обоих монастырей собираются на совместное празднество и показывают свое совершенствование на Пути Кулака. Всех соревнующихся монахов разбивают на пары, победители пар бьются затем между собой и так далее, до финального поединка. Монастырь, монах которого победил в финальном бою, забирает себе на хранение статую боддисатвы. Реализовать многопоточное приложение, определяющего победителя. В качестве входных данных используется массив, в котором хранится количество энергии Ци каждого монаха. При решении использовать принцип дихотомии.

Лаб 3
Тема:Двоичные семафоры: защита совместного доступа к памяти

а) **Задача о Винни-Пухе или правильные пчелы.** В одном лесу живут n пчел и один медведь, которые используют один горшок меда, вместимостью H глотков. Сначала горшок пустой. Пока горшок не наполнится, медведь спит. Как только горшок заполняется, медведь просыпается и съедает весь мед, после чего снова засыпает. Каждая пчела многократно собирает по одному глотку меда и кладет его в горшок. Пчела, которая приносит последнюю порцию меда, будит медведя. Создать многопоточное приложение, моделирующее поведение пчел и медведя.

б)**Задача о парикмахере.** В тихом городке есть парикмахерская. Салон парикмахерской мал, ходить там может только парикмахер и один посетитель. Парикмахер всю жизнь обслуживает посетителей. Когда в салоне никого нет, он спит в кресле. Когда посетитель приходит и видит спящего парикмахера, он будет его, садится в кресло и спит, пока парикмахер занят стрижкой. Если посетитель приходит, а парикмахер занят, то он встает в очередь и засыпает. После стрижки парикмахер сам провожает посетителя. Если есть ожидающие посетители, то парикмахер будит одного из них и ждет пока тот сядет в кресло парикмахера и начинает стрижку. Если никого нет, он снова садится в свое кресло и засыпает до прихода посетителя. Создать многопоточное приложение, моделирующее рабочий день парикмахерской.

с) **Задача о курильщиках.** Есть три процесса-курильщика и один процесс-посредник. Курильщик непрерывно скручивает сигареты и курит их. Чтобы скрутить сигарету, нужны табак, бумага и спички. У одного процесса- курильщика есть табак, у второго – бумага, а у третьего – спички. Посредник кладет на стол по два разных случайных компонента. Тот процесс-курильщик, у которого есть третий компонент, забирает компоненты со стола, скручивает сигарету и курит. Посредник дожидается, пока курильщик закончит, затем процесс повторяется. Создать многопоточное приложение, моделирующее поведение курильщиков и посредника. При решении задачи использовать семафоры.

Лаб4
Тема:Блокировки чтения-записи
а) **Задача о читателях и писателях.** Базу данных разделяют два типа процессов – читатели и писатели. Читатели выполняют транзакции, которые просматривают записи базы данных, транзакции писателей и просматривают и изменяют записи.
Создать многопоточное приложение, работающее с общим файлом.
Для защиты операций с общим файлом использовать блокировки чтения-записи. Файл содержит последовательность записей вида: Ф.И.О.1 – телефон1, Ф.И.О.2 – телефон2… В приложении должны работать следующие потоки:

1) потоки, находящие телефоны по указанной фамилии;
2) потоки, находящие Ф.И.О. по указанному телефону;
3) потоки, удаляющие и добавляющие записи в файл.

б) **Задача про сад.** Создать многопоточное приложение, работающее с общим двумерным массивом. Для защиты операций с общим массивом использовать блокировки чтения-записи. Двумерный массив описывает сад. В приложении должны работать следующие потоки:
1) поток-садовник следит за садом и поливает увядшие растения;
2) поток-природа может произвольно изменять состояние растений;
3) поток-монитор1 периодически выводит состояние сада в файл (не стирая предыдущее состояние);
4) поток-монитор2 выводит состояние сада на экран.

с) **Задача про автобус**. Создать многопоточное приложение, работающее с общим графом. Для защиты операций с графом использовать блокировки чтения-записи.
Граф описывает множество городов и множество рейсов автобусов от города А к городу Б с указанием цены билета (по умолчанию, если рейс от А к Б, он идет и от Б к А, с одинаковой ценой). В приложении должны работать следующие потоки:
1) поток, изменяющий цену билета;
2) поток, удаляющий и добавляющий рейсы между городами;
3) поток, удаляющий старые города и добавляющий новые;
4) потоки, определяющие есть ли путь от произвольного города А до произвольного города Б, и какова цена такой поездки (если прямого пути нет, то найти любой путь из существующих)

Лаб 5
Тема: барьеры
а)**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 многопоточное 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.

б) Создать приложение с четырьмя потоками. Каждый поток работает с собственной строкой. Строки могут содержать только символы A, B, C, D. Поток может поменять символ A на C или C на A или B на D или D на B. Потоки останавливаются когда общее количество символов A и B становится равным хотя бы для трех строк.

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

Лаб 6

Клеточный автомат: игра "Жизнь"

Основная идея — разделить пространство физической или биологической задачи на отдельные клетки. Каждая клетка — это конечный автомат. После инициализации все клетки сначала совершают один переход в новое состояние, затем второй переход и т.д. Результат каждого перехода зависит от текущего состояния клетки и ее соседей.

Дано двухмерное поле клеток. Каждая клетка либо содержит организм (жива), либо пуста (мертва). В этой задаче каждая клетка имеет восемь соседей, которые расположены сверху, снизу, слева, справа и по четырем диагоналям от нее. У клеток в углах по три соседа, а на границах — по пять.

Игра "Жизнь" происходит следующим образом. Сначала поле инициализируется. Затем каждая клетка проверяет состояние свое и своих соседей и изменяет свое состояние в соответствии со следующими правилами.

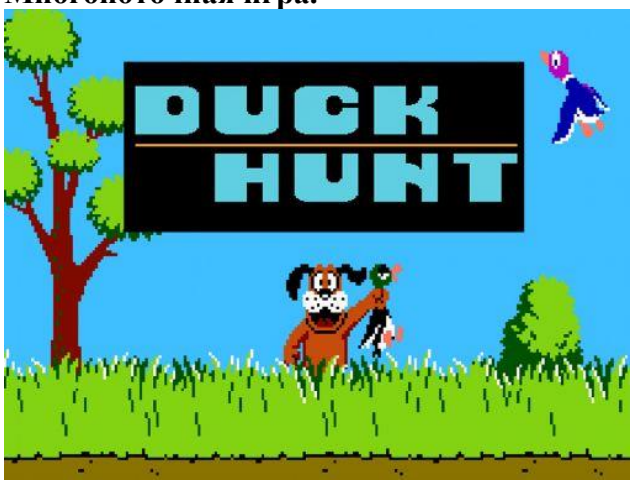
- *Живая клетка, возле которой меньше двух живых клеток, умирает от одиночества.*
- *Живая клетка, возле которой есть две или три живые клетки, выживает еще на одно поколение.*
- *Живая клетка, возле которой находится больше трех живых клеток, умирает от перенаселения.*
- *Мертвая клетка, рядом с которой есть ровно три живых соседа, оживает.*

Этот процесс повторяется некоторое число шагов (поколений).
См. также http://life.written.ru/game_of_life_review_by_gardner

а) Предусмотреть развитие цивилизации (живые клетки) в нескольких потоках, объяснить выбранный вариант эфективного распределения клеток по потокам

б) Одновременно стартовать несколько цивилизаций разными цветами при конкуренции за ресурсы (клетки) использовать атомарные операции, семафоры или мониторы, при изменении состояния клетки.

Лаб7
Многопоточная игра.



Ориентировочный вариант игры: https://www.youtube.com/watch?v=nnEhEKR_Rs4
а) Утки(несколько с разной скоростью и направлением) в отдельных потоках. Сбивание происходит по нажатию кнопки мыши по утке.
б) Дополнительно поток с Охотником который двигается в нижней части при выстреле(пробел) вылетает пуля. Сбивание происходит при пересечении пули с объектом Утки.

Лаб8

Вычисления на кластере с использованием MPI.
Создать приложение в котором вычисляется параллельное умножение матриц.

- Последовательный алгоритм
- Алгоритм1 –ленточная схема
- Алгоритм2 –метод Фокса
- Алгоритм3 –метод Кэннона

Расчеты по каждому алгоритму сделать ввиде таблицы последовательный(1 поток), 2 потока, 4 потока.

розмірність	**послідовний (1потік),час**	**2 потоки**			**4 потоки**		
		час	Sp	Ep	час	Sp	Ep
100							
1000							
5000							

Лаб9
Написать программу из **Лаб8(Алгоритм1 –ленточная схема)** используя функции позволяющими автоматически распараллеливать код(замерить время выполнения):

1. java.util.concurrent. ForkJoinPool
   http://www.h-online.com/developer/features/The-fork-join-framework-in-Java-7-1762357.html
   http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/package-tree.html
   http://www.oracle.com/technetwork/articles/java/fork-join-422606.html
2. Multi-Core Parallel Programming in **Go (GO)**
   https://golang.org/doc/effective_go.html#concurrency
   https://www.youtube.com/watch?v=f6kdp27TYZs
   https://www.youtube.com/watch?v=QDDwwePbDtw
3. **Intel Threading Building Blocks** для разработки многопоточного ПО
   https://www.threadingbuildingblocks.org/docs/help/tbb_userguide/
4. OpenMP параллельный регион #pragma omp parallel for
   https://software.intel.com/ru-ru/articles/getting-started-with-openmp
5. TPL (C#) параллельный цикл Parallel.for;
   https://msdn.microsoft.com/ru-ru/library/dd460717%28v=vs.110%29.aspx