Лабораторная работа № 3

Многопоточное обслуживание клиентов при организации распределённой обработки информации средствами стека протоколов TCP/IP.

1. Теоретические сведения

Пул потоков в ОС Windows.

Для организации пула потоков в ОС Windows используется функция QueueUserWorkItem

BOOL QueueUserWorkItem(PTHREAD_START_ROUTINE pfnCallback, PVOID pvContext,ULONG dwFlags);

Эта функция помещает "рабочий элемент" (work item) в очередь потока в пуле и тут же возвращает управление.Рабочий элемент — это просто вызов функции(на которую ссылается параметр pfnCallback), принимающей единственный параметр, pvContext.В конечном счете какой - то поток из пула займется обработкой этого элемента, в результате чего будет вызвана Ваша функция.У этой функции обратного вызова, за реализацию которой отвечаете Вы, должен быть следующий прототип :

DWORD WINAPI WorkItemFunc(PVOID pvContext);

Несмотря па то, что тип возвращаемого значения определен как DWORD, на самом деле оно игнорируется.

Обратите внимание, что Вы сами никогда не вызываете CreateThread.Она вызывается из пула потоков, автоматически создаваемого для Вашего процесса, а к функции WorkItemFunc обращается один из потоков этого пула.Кроме того, данный поток не уничтожается сразу после обработки клиентского запроса, а возвращается в пул, оставаясь готовым к обработке любых других элементов, помещаемых в очередь.Ваше приложение может стать гораздо эффективнее, так как Вам больше не придется создавать и уничтожать потоки для каждого клиентского запроса.А поскольку потоки связаны с определенным портом завершения, количество одновременно работающих потоков не может превышать число процессоров более чем в 2 раза.За счет этого переключения контекста происходят реже.

Многое в пуле потоков происходит скрытно от разработчика: QueueUserWorkItem проверяет число потоков, включенных в сферу ответственности компонента поддержки других операций(нс относящихся к вводу - выводу), и в зависимости от текущей нагрузки (количества рабочих элементов в очереди) может передать ему другие потоки.После этого QueueUserWorkItem выполняет операции,

эквивалентные вызову PostQueuedCompletionStatus, пересылая информацию о рабочем элементе в порт за вершения ввода - вывода.В конечном счете поток, ждущий на этом объекте, извлекает Ваше сообщение(вызовом GetQueuedCompletionStatus) и обращается к Вашей функции. После того как она возвращает управление, поток вновь вызывает GetQueuedComplettonStatus, ожидая появления следующего рабочего элемента. Пул рассчитан на частую обработку асинхронного ввода - вывода — всякий раз, когда поток помещает в очередь запрос на ввод - вывод к драйверу устройства.Пока драйвер выполняет его, поток, поставивший запрос в очередь, не блокируется и может заниматься другой работой.Асинхронный ввод - вывод — ключ к созданию высокоэффективных, масштабируемых приложений, так как позволяет одному потоку обрабатывать запросы от множества клиентов по мере их поступления; ему не приходится обрабатывать их последовательно или останавливаться, ожидая завершения ввода - вывода.

Но Windows накладывает одно ограничение на запросы асинхронного ввода - вывода, если поток, послав такой запрос драйверу устройства, завершается, данный запрос теряется н никакие потоки о его судьбе не уведомляются.В хорошо продуманном пуле, число потоков увеличивается и уменьшается в зависимости от потребностей его клиентов.Поэтому, если поток посылает запрос и уничтожается из - за сокращения пула, то уничтожается и этот запрос.Как правило, это не совсем то, что хотелось бы, и здесь нужно найти какое - то решение.

Если Вы хотите поместить в очередь рабочий элемент, который выдает запрос на асинхронный ввод - вывод, то не сможете передать этот элемент компоненту поддержки других операций в пуле потоков.Его примет лишь компонент поддержки ввода вывода.

Последний включает набор потоков, которые не завершаются, пока есть хотя бы один запрос на ввод - вывод; поэтому для выполнения кода, выдающего запросы на асинхронный ввод - вывод, Вы должны пользоваться только этими потоками.

Пример пула потоков ОС Windows.

```
#include <windows.h>
DWORD WINAPI ThreadFunc(LPVOID context)
{
	HANDLE hEvent = (HANDLE)context;
	printf("Hello world from worker thread\n");
	SetEvent(hEvent);
	return 0;
}
#pragma argsused
int main(int argc, char* argv[])
{
	hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
	QueueUserWorkItem(ThreadFunc, (PVOID)hEvent, WT_EXECUTEDEFAULT);
	WaitForSingleObject(hEvent, 1000);
	CloseHandle(hEvent);
```

```
    return 0;
}
```

В ОС Linux отсутствуют стандартные средства реализации пула потоков. Однако он может быть реализован самостоятельно.

2. Задание на лабораторную работу.

Задания представленные ниже выполняются с использованием WinAPI и WSA Socket или API Linux.

Варианты заданий

	4-5 баллов
	6-8 баллов
	9-10 баллов

Разработать сервер с использованием пула потоков для параллельной обработки запросов клиентов. К серверу подключается много клиентов, все они отправляют задачи (по вариантам), сервер обрабатывает каждый запрос в отдельном потоке (используя пул потоков) и возвращает ответ клиенту.

Обязательные компоненты отчета:

- Краткое описание алгоритма
- Верификация

Варианты согласуются с преподавателем:

1.	Вычисление псевдослучайного числа используя регистр сдвига с линейной обратной связью (LFSR)
2	Одномерная оптимизация функции методом деления попалам
3	Mergesort
4	Одномерная оптимизация функции методом золотого сечения
5	Quicksort
6	Heapsort
7	Insertion Sort
8	Одномерная оптимизация функции методом равномерного поиска
9	Умножение матриц
10	Одномерная оптимизация функции методом Фибоначчи

11	Treesort
12	Одномерная оптимизация функции методом касательных
13	Разбор математического выражения и вычисление с учетом скобок и приоритетов операций (+ - * /) . Использовать обратную польскую запись
14	Вычисление секанса (без использования математических библиотек, используя разложение в ряд Маклоррена)
15	Вычисление синуса (без использования математических библиотек, используя разложение в ряд Маклоррена)
16	Вычисление экспоненты (без использования математических библиотек, используя разложение в ряд Маклоррена)
17	Разбор математического выражения с учетом приоритетов операций (+ - * /), и тригонометрических функций (sin, cos, tg) . Использовать обратную польскую запись.
18	Вычислить число ПИ до 50 знака методом Монте-Карло
19	Вычислить число ПИ до 50 знака методом иглы Буффона
20	Вычислить число ПИ до 50 знака разложением в ряд Нилаканта
21	Вычислить квадратный корень из числа, используя итерационную формулу Герона
22	Вычислить квадратный корень из числа, используя алгоритм Ньютона
23	Вычисление натурального логарифма (без использования математических библиотек, используя разложение в ряд Маклоррена)
24	Вычисление гиперболического синуса(без использования математических библиотек, используя разложение в ряд Маклоррена)
25	Вычислить число ПИ до 50 знака разложением в ряд Лейбница
26	Решение СЛАУ через обратную матрицу
27	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность вызовов процедур, разделенных символом ;(точка с запятой). Вызов процедуры должен состоять из имени процедуры и списка параметров. В качестве параметров могут выступать идентификаторы, строковые константы, заключенные в двойные кавычки и одиночные символы, заключенные в одинарные кавычки. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.

28	Разработать конечный автомат для анализа вырожения. Входной язык содержит арифметические выражения, разделенные символом ;(точка с запятой). Арифметические выражения состоят из идентификаторов, римских чисел, знаков операций и скобок. (Римскими считать числа записанные большими буквами X, V и I). Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
29	Разработать конечный автомат для анализа вырожения. Входной язык содержит HTML код, нужно проверить правильность объявления ссылки. Учитывать что могут быть кавычки двух типов или отсутствовать. Проверка атрибутов href, alt, blank. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
30	Разработать конечный автомат для анализа вырожения. Входной язык содержит HTML код, нужно проверить правильность тега img. Учитывать что могут быть кавычки двух типов или отсутствовать. Проверка атрибутов src, alt, width, height. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
31	Разработать конечный автомат для анализа корректности email. Найти все email в файле отправленном пользователем. Клиенту возвращается список всех найденных email в тексте
32	Найти обратную матрицу (используя метод Гаусса)
33	Разбор математического выражения и вычисление с учетом скобок и приоритетов операций (+ - * /) . Использовать дерево выражений
34	Граф хранится на сервере, пользователь указывает название точки отправления и точки назначения, в ответ получает кратчайший путь. Для поиска пути использовать Алгоритм Флойда — Уоршелла
35	Граф хранится на сервере, пользователь указывает название точки отправления и точки назначения, в ответ получает кратчайший путь. Для поиска пути использовать Алгоритм Дейкстры

36	Граф хранится на сервере, пользователь указывает название точки отправления и точки назначения, в ответ получает кратчайший путь. Для поиска пути использовать Алгоритм Беллмана — Форда
37	Решение СЛАУ методом Гаусса
38	Решение СЛАУ методом Жардана-Гауса
39	Найти определитель матрицы
40	LDL^T разложение матрицы
41	Найти обратную матрицу (через единичную)
42	Разложение Холецкого
43	LU разложение матрицы
44	Умножение матриц алгоритмом Штрассена
45	Шифрование строки алгоритмом RSA
46	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность описаний массивов в соответствии со спецификацией языка C#, разделенных символом ;(точка с запятой). Считать, что массивы могут содержать только элементы следующих типов: int, double, float, char и String. Массивы инициализируют только фиксированной длины. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
47	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность команд ассемблера в форме: <метка>: <команда> <операнд1>,<операнд2> (метка, а также один или оба операнда могут отсутствовать). В качестве операндов могут выступать регистры процессора 80x86, идентификаторы, целые десятичные числа или целые шестнадцатеричные числа. (Предусмотреть наличие не менее 6 допустимых команд). Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
48	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность описаний записей (record) в соответствии со спецификацией языка Паскаль,

	разделенных символом ;(точка с запятой). Считать, что записи могут содержать только поля скалярных типов integer, real, byte, word, char и строки string с возможным указанием длины строки в квадратных скобках. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
49	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность описаний массивов в соответствии со спецификацией языка Паскаль, разделенных символом ;(точка с запятой). Считать, что массивы могут содержать только элементы скалярных типов integer, real, byte, word и char. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
50	Разработать конечный автомат для анализа вырожения. Входной язык содержит последовательность вызовов процедур, разделенных символом ;(точка с запятой). Вызов процедуры должен состоять из имени процедуры и списка параметров. В качестве параметров могут выступать идентификаторы, целые десятичные числа без знака, шестнадцатеричные числа, десятичные числа с плавающей точкой. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
51	Разработать конечный автомат для анализа вырожения. Входной язык содержит выражения над строковыми константами, разделенные символом ;(точка с запятой). Выражения состоят из идентификаторов, строковых констант, заключенных в двойные кавычки, одиночных символов, заключенных в одинарные кавычки и знаков операции конкатенации +. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
52	Разработать конечный автомат для анализа вырожения. Входной язык содержит упрощенные условные операторы типа

	if <логическое выражение> then <оператор присваивания> else <оператор присваивания>; (часть else в операторе может отсутствовать). Логическое выражение может содержать идентификаторы, знаки операций сравнения, целые десятичные числа без знака, скобки и логические операции and и not. Оператор присваивания должен состоять из двух идентификаторов, разделенных знаком присваивания. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
53	Разработать конечный автомат для анализа вырожения. Входной язык содержит упрощенные операторы цикла типа while <логическое выражение> do <оператор присваивания>; Логическое выражение может содержать идентификаторы, знаки операций сравнения, целые десятичные числа без знака, скобки и логические операции and и or. Оператор присваивания должен состоять из идентификатора, знака присваивания и целой десятичной константы без знака. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
54	Разработать конечный автомат для анализа вырожения. Входной язык содержит арифметические выражения, разделенные символом ;(точка с запятой). Арифметические выражения состоят из идентификаторов, десятичных чисел с плавающей точкой (в обычной и логарифмической форме), знаков операций и скобок. Клиенту возвращается удовлетворяет ли переданное выражение заданному языку или нет. Обработку запросов от клиентов реализовать с использованием пула потоков.
55	решения систем обыкновенных дифференциальных уравнений Рунге-Кутты 4-го порядка
56	решения систем обыкновенных дифференциальных уравнений методом Эйлера
57	Разработать конечный автомат для анализа корректности номера телефона в формате.[+XXX (YY) XXX-XX-XX] Найти все номера телефонов в файле отправленном пользователем. Клиенту возвращается список всех найденных номеров телефонов в тексте
58	Разработать конечный автомат для анализа надежности и корректности пароля

	Длина пароля не менее 6 символов. Разрешены только латинские буквы, цифры и знаки: пробел _ - . В пароле обязательно должны присутствовать знаки в верхнем и нижнем регистре и цифры. Пользователю возвращается информация о том, можно ли использовать указанный пароль
59	Медианный фильтр изображения с возможностью выбора размера окна
60	Фильтр Гаусса для изображений с возможностью выбора размера окна
61	Фильтр Собеля (оператор Собеля) для выявления границ объектов изображения
62	Перекрестный оператор Робертса для выявления границ объектов на изображении
63	Оператор Прюитт для выделения границ на изображении
64	Проверка вхождения двух человек в одну группу. Исходные данные, имена людей, которые добавлены в друзья в социальной сети. Пользователь вводит 2 имени, нужно определить связаны ли они через общих друзей. (Find Union)
65	Компрессия/Декомпрессия данных. Алгоритм Лемпеля — Зива — Велча (LZW)
66	Компрессия/Декомпрессия данных. алгоритмом Хаффмана
67	Многомерная оптимизация функции методом деформируемого многогранника (Нелдера-Мида)
68	Многомерная оптимизация функции методом Хука Дживса
69	Многомерная оптимизация функции методом градиентного спуска (построить график)
70	Оператор Кенни для поиска границ объектов на изображении

3. Контрольные вопросы

1. Пул потоков
2. Преимущества и недостатки пула потоков
3. Реализация пула потоков в ОС Windows
4. Как можно реализовать пул потоков в ОС Linux
5. Функция QueueUserWorkItem
6. Класс ThreadPool
7. Использование пула потоков в сетевом приложении для обработки запросов от клиентов