Задачи для 4 семестра 2 курса (2025)

Целью работы в данном семестре является более глубокое знакомство с возможностями языка C++, а также элементарное введение в специфику сетевого программирования и WEB интерфейсов.

При выполнении задания предполагается использование существующих возможностей языка C++ таких как, например, классов библиотеки STL, умных указателей, семантики перемещения и других инструментов современного программирования на C++.

Отдельной особенностью работы в семестре является моделирование "производственного режима программирования", когда в требования к реализации постоянно вносятся дополнительные изменения, а сама реализация постоянно модифицируется. Для поддержания работоспособности такой программы надо заранее уделять особое внимание "архитектуре" всего проекта и структуре программного кода, чтобы сделать изменение программы более простым и облегчить обнаружение и исправление ошибок.

Кроме этого также предполагаются самостоятельные и контрольные работы, не связанные непосредственно с основными заданиями и проверящих умение реализовывать некоторые "стандарные" схемы обработки данных.

Все "теоретические" материалы, в частности, принципы сетевого программирования, примеры реализаций UDP и TCP клиентов и серверов, упомянутые выше инструменты C++ будут обсуждаться на лекциях.

Основное задание

Задание состоит в реализации "большого проекта", который включает в себя модельную базу данных с некоторым языком запросов, реализацию этой базы в виде сетевой модели клиент-сервер, поддержка WEB интерфейса для работы с этой базой через WEB браузер по стандарту протокола HTTP, и другие компоненты, исходя из специфики конкретной задачи. Кроме этого предполагается знакомство техникой работы с большими проектами — использование систем сборки (make, CMake), использование внешних библиотек, работа с другими языками (Python) и т.п.

Задачи такого типа весьма многогранны, поэтому варианты заданий преполагают некоторую общую часть, которая затем будет модифицироваться и дополняться по индивидуальным дополнительным заданиям.

Работа с базой данных предполагает действия в режиме "запрос–ответ". Это означает, что клиент формирует некоторый запрос на требуемое действие (модификация базы данных, запрос на выборку данных по указанным критериям и т.п.), а сервер базы выполняет эти действия и отправляет клиенту результат в той или иной форме. Это процесс продолжается в рамках установленного сеанса работы с базой данных. Отметим, что в рамках распределенного режима работы к обной и той же базе (серверу) могут одновременно обращаться несколько независимых клиентов. Таким образом, одной из возможных проблем при реализации является поддержка корректности обращений и результатов, чтобы клиенты "не мешали" друг другу и база в целом оставалась в корректном состоянии.

Таким образом, выполнение задания разбивается на несколько этапов, каждый из которых отвечает за свою сторону работы всего проекта и может и должен выливаться в реализацию некоторого набора программ (модулей), составляющих проект. При этом по возможности реализация каждого такого модуля должна быть пробумана так, чтобы их можно было модифицировать независимо друг от друга с сохранением работоспособноси проекта в целом.

ВАЖНО ! При проверке этапов выполнения задания необходимо четко соблюдать требования к выполнению этапа. А именно, — к каждому этапу предоставлется отчет, который должен содержать информацию о том, что конкретно реализовано на данном этапе, какие структурные и программные решения реализованы, какие рассматривались, но не были реализованы и почему;

— как выполнялось тестирование этапа, т.е. какие задачи ставились перед тестированием, что и каким образом проверялось, возможности конкретной версии программы (что реализовано, а что только планируется), ограничения конкретной версии, и каким образом при желании можно изменить тестовые данные и т.п.;

— работоспособность программы проверяется в OS Linux со строгими настройками компилятора g++, соответственно, при ошибках компиляции текущий этап не принимается;

— приветствуется использование процедур сборки проектов типа make или CMake (чуть позже это требование станет обязательным). Таким образом, при правильно подготовленном этапе проверка задания будет состоять в чтении отчета и запуске программы для проверки ее работоспособности. От качества и полноты вашего собственного тестирования этапа и будет зависеть результат приема задания.

Общие этапы и соответствующие контрольные сроки по заданию выглядят примерно так: Всего у нас получается 14 недель

1 этап (2 недели) Выбор предметной области базы данных. Разработка структуры записей базы данных. Описание множества допустимых операций (запросов). Подготовка (генерация) файла с исходными данными (записями базы). Примеры возможных предметных областей см. далее.

Мы предполагаем, что база, как правило, представляет собой набор записей некоторй фиксированной структуры, отражающей необходимую пользователю информацию. Для отладки и работы с базой необходим содержательный набор таких записей. Поэтому на данном этапе требуется реализовать генератор исходного файла записей базы данных по теме своей задачи. Генератор должен порождать большое количество записей осмысленного вида для дальнейшего тестирования базы. Например, имена и фамилии должные быть реальными, числовые или календарные характеристики должны находиться в требуемом диапазоне значений. Как правило, генератор должен позволять создание базы на очень большое число записей (миллионы).

Отчет по данному этапу состоит в демонстрации возможностей генератора:

— описание формата записи базы и смысла ее отдельных полей;

— описание возможностей и ограничений генератора (насколько разнообразные записи он может порождать);

— описание как сгенерировать файл на заданное количество записей;

— пример сгенерированного файла на некоторое количество записей.

— комментарии к коду представления данных, какие классы за что отвечают и для чего предназначены (также в коде приветствуются комментарии по назначению классов и их компонент).

2 этап (2 недели) Реализация работы с базой данных (загрузка, поиск, добавление, удаление записей) на основе простейшего внутреннего представления базы в виде линейного массива записей. Программа загружает записи из файла. При необходимости проверяет их корректность. Обеспечивает выполнение требуемых операций через простейший консольный интерфейс.

На данном этапе уже важно разделить функции хранения базы данных и операций с ней и функций организации обращения к базе данных и обработки полученного результата.

При выполнении этого этапа необходимо определиться со следующими решениями:

— какие операции допускаются с базой (выборка, добавление, удаление, модификация записей, вывод выборки и т.д.);

— по каким полям записей будет осуществляться выборка;

— какие условия будут участвовать в критериях выборки каждого поля (равенство, неравенство, диапазон, включение и т.п.);

— какими параметрами можно описать условия для проверки конкретной записи на принадлежность к выборке;

— в какой форме будет представлена выборка в вашей базе (например, индексный массив, массив указателей и т.п.);

— сохранение измененной базы между сеансами работы с ней.

Как вариант, данный этап может заключаться в создании некоторого количества “интерфейсных” функций для поиска записей, удовлетворяющих заданным критериям, для изменения записей, для загрузки и сохренения базы и т.п. с определеной системой диагностики успешности выполняемых операций.

Очень важным является решения вопроса о том, что будет представлять собой выборка, сформированная по результату запроса. На мой взгляд, наиболее эффективным решением является массив, в котором собраны идентфикаторы (индексы, указатели) соответствующих записей базы данных. В этом случае повторный анализ данной выборки (если будет нужно) не потребует прохода по всей базе данных, а лишь только по уже отобранным записям.

Для интерфейсной функции, обрабатывающей запрос выборки, будет удобно задать полный набор критериев отбора по всем рассматриваемым полям (границы значений полей, списки перечисления значений и т.д.). Тогда любой запрос будет сводиться к вызову этой функции со специфичным набором параметров, а конкретный алгоритм поиска будет уже в зоне ответственности этой функции.

Отчет по данному этапу состоит из

— описания принятых решений по набору операций (см. выше);

— описания интерфейса и комментарии по поводу функций, выполняющих основные операции с базой (выборка, добавление и т.д.) т.е. какие у них заголовки и параметры, что эти параметры означают, где в коде эти фукции находятся;

— тестовый сценарий работы с базой — серия вызовов функций, реализующих операции с базой, и пояснения почему результат будет правильный;

Нет необходимости реализовывать консольный диалог ввода параметров поиска или других операций с базой. Более наглядно подготовить набор тестовых вызовов своих интерфейсных функций в разной последовательности и с разными наборами параметров для иллюстрации работы. Все равно впоследствии подобный диалог не будет нужен, а вызовы будут выполняться на основе разбора языка запросов.

3 этап (2 недели) Разработка и реализация языка запросов к базе для операций выборки, добавления, удаления. Запрос — это текстовая строка, в которой в некоторой относительно свободной форме записаны необходимые параметры. Например, для базы студентов
select group=201-226 rating<3 end
select name=Петров|Petrov group>200 info "общежитие" end
Содержание и поля запроса определяются конкретной задачей и сценарием работы. Аналогично могут выглядеть запросы add, remove, edit и другие подобные. Сложность конструкций запросов также может варьироваться в зависимости от содержаний задачи.

Ответом на запрос select является выборка, т.е. набор записей. Возможно, пользователю интересны не все поля из этих записей, а только некоторые и саму выборку пользователь может хотеть видеть отсортированной по некоторому принципу. В таком случае можно также предусмотреть запрос print, sort, который будет определять форму выдачи и сортировку.

По содержанию данный этап состоит в анализе строки запроса и извлечении из нее параметров для обращения к процедурам работы с базой из предыдущего этапа (выборка, добавление, удаление и пр.). При этом для параметров, отсутствующих в запросе, можно предусмотреть некие "общие" значения типа бесконечного диапазона значений, произвольного включения и т.п. с тем, чтобы при обращении к функциям выборки эти параметры не влияли на результат.

На этом этапе имеет смысл предусмотреть возможность считывать запросы из текстового файла и вывод соответствующих ответов также в файл. Тем самым будет подготовлена почва для последующей автоматизации работы клиентов.

Отчет по данному этапу состоит из

— описания формата запросов и записи возможных критериев выборки;
— тестирующей процедуры, которая считывает из файла запросы и выводит ответы в файл;
— несколько файлов с серией запросов, которые используются этой тестирующей процедурой;
— пояснения какие тесты выполнялись и почему они считаются пройденными успешно.

4 этап (2 недели) Создание сетевой реализации с разделением на клиентскую и серверную часть. Разные варианты сетевого взаимодействия проиллюстрированы имеющимися программными заготовками. Надо в них разобраться и модифицировать под свои нужды. При этом надо понимать все варианты, чтобы можно было без проблем модифицировать реализации на разные протоколы и разные способы распределения обработки клиентов. Проверка работоспособности в ручном режиме, т.е. клиент активируется пользователем. Здесь надо также обратить внимание на вопросы блокировки базы данных при обработке редактирующих запросов с тем, чтобы не разрушить ее целостность при одновременных обращениях к одним и тем же записям.

По сути нужно взять тестирующую процедуру из предыдущего этапа и разделить ее на две части. Часть, которая читает запросы из файла составит содержательную часть клиента. Затем клиент должен просто отправиль эти запросы в виде текстовых строк на сервер. Оставшаяся часть реализации (разбор запроса, выделене параметров выборки, вызов функции выборки) составит содержательную часть серверной базы. После выполнения выборки сервер должен будат отправить записи выборки в текстовом виде обратно клиенту, а клиент, соответственно, их принять и сделать то, что от него требуется (распечатать, сохранить в файл и т.п.).

Основная проблема этого этапа — реализация асинхронного обмена данными между клиентом и сервером и необходимость приема данных разной длины (особенно, на стороне клиента). Поэтому здесь надо предусмотреть дополнительные сообщения между клиентом и сервером, чтобы они могли согласовать размеры буферов приема и передачи сообщений и информировали друг друга о количестве сообщений и их длине. Т.е. фактически разработать некоторый протокол работы с удаленной базой.

Тест дожен проверять отправку серии запросов от клиента на сервер и получении ответов (запросы читаются из файла, ответы сохраняются в файле). Следует проверить как клиант принимает ответы существенно разного объема.

Отчет по данному этапу состоит из

— описания структуры реализации клиента и сервера, какие сетевые протоколы выбраны, какие способы передачи данных и проверки каналов обмена использованы;
— описание логики взаимодействия клиента и сервера, какими сообщениями они обмениваются при обработке запроса и отправки ответа;
— описание содержания теста, что делается, какой сценарий реализуется;
— описание того как запустить тест на выполнение и как убедиться, что тест прошел успешно.

5 этап (2 недели) Автоматизация клиентской части. Создание клиентов-генераторов для формирования непрерывного потока запросов к базе на основе разных сценариев. Фактически этот этап уже должен быть реализован на предыдущем шаге. Здесь предполагается, что создаются разнообразные файлы со сценариями запросов и указанием интервалов по времени, чтобы можно было смоделировать "независимую" и несогласованную работу нескольких клиентов с одним сервером. Здесь следует предусмотреть возможность идентификации клиентов и распечатку мо-

ментов времени, чтобы потом можно было проследить хронологию обрабоки запросов и запуск нескольких клиентов с помощью командных файлов.

Основным механизмом проверки асинхронного взаимодействия является моделирование случайных моментов отправления запросов и получения ответов с условиях работы нескольких клиентов. Это достигается вставкой задержки на заданный или случайный интервал времени с помощью функций типа sleep или usleep. Вызовы этих функций вставляются между отправкой отдельных запросов каждого клиента и между отправкой отдельных ответных сообщений сервера.

Тест состоит в запуске нескольких клиентов, каждый со своим сценарием запросов и задержек (задаются в файле). Результаты ответов сервера для каждого отдельного клиента не должны зависеть от количества других клиентов и от сценариев их работы.

Для запуска такого теста можно подготовить командный файл запуска нескольких клиентов, и каждому клиенту передавать через командную строку (параметры функции main) имена файлоа со сценарием запросов и файла для сохранения ответа.

Отчет по данному этапу состоит из

— описания механизма отработки сценариев задержки на стороне клиента и на стороне сервера, т.е. в каких местах кода вставляются задержки, по какой логике выбираются интервалы времени для задержки;

— описание файлов со сценариями запросов и прочая информация, необходимая для запуска теста;

— описание вариантов теста и ожидаемых результатов.

6 этап (2 недели) Реализация HTTP протокола (частичная) на стороне сервера. Использование стандартного интернет браузера в качестве клиента базы данных. Разработка и реализация соответствующих HTML форм для ввода запросов и таблиц для вывода результатов выборки.

Клиент заменяется стандартным интернет браузером. Это означает, что браузер обращается к предваритльно запущенному серверу, запросы вводятся пользователем в окне браузера, а ответы сервера (выборки) выводятся также в окне браузера.

Следует разобраться в минимальных и простейших принципах работы HTTP протокола для правильного разбора запросов, приходящих от браузера, и для правильного формирования ответов сервера, чтобы браузер мог фдекватно отобразить присланную информацию.

Кроме этого надо познакомится с основными принципами записи HTML файлов, в частности, с отображением текста, строеним таблиц и понятием формы и полей ввода.

Отчет по данному этапу состоит из

— описание того, как запустить сервер и выполнить соединение браузера с этим сервером;

— описание того где и в какой форме пользователь может вводить запросы к базе данных;

— примеры запросов, которые использовались для тестирования и характеристика полученных результатов.

Выше описаны базовые требования к реализации проекта. Возможны и рекомендуются индивидуальные модификации заданий в направлении усложнения самых примитивных решений. Например, последовательный поиск записей при выборке может быть дополнен поиском по деревьям (с соответствующей модификацией сжемы хранения данных). Язык запросов может получить более сложный синтаксис с логическими выражениями для критериев выборки. Сетевая часть может быть реализована в разных схемах и протоколах. Для отображения в браузере могут быть использованы более сложные конструкции HTML и т.п. Способность студента вносить подобные локальные модификации в свой проект свидетельствуют о грамотной разработке архитектуры проекта.

Каждый из 6 этапов предполагает отдельное оценивание. Т.е. к концу семестра у каждого должно накопиться некоторое количество “плюсов” (или “минусов”) за эти этапы и еще плюсы/минусы за самостоятельные и контрольные работы. Пропуск контрольного срока сдачи этапа выливается в появление “минуса”, который придется превращать в плюс дополнительными (бонусными) усилиями.

Самостоятельные и контрольные работы. Поскольку преполагается знакомство с дополнительными возможностями языка C++ (умные указатели, семантика перемещения, многопоточность и др.), то контрольные работы могут содержать примеры на проверку знания этих техник.

Кроме этого предполагаются задачи на освоение простейшего сетевого программирования. Здесь предполагаются задания, не требующие особой алгоритмической изощренности, а просто на понимание принципов асинхронного сетевого взаимодействия. Конкретные формулировки могут меняться, а в идейном плане можно рассматривать взаимодействия примерно такого вида:

Сервер-вычислитель. Выполняет простейшие вычисления по запросу клиента и возвращает ответ. Возможен вариант с циклической работой с каждым клиентом до выполнения некоторого условия.

Аукцион. Сервер принимает “ставки” от клиентов и определет “победителя” по достижении заданного условия (например, момента времени).

Почтовый сервер. Регистрирует обратившихся клиентов и пересылает сообщения между зарегистрированными клиентами.

Сервер новостей. Периодически генерирует “новости” и рассылает их всем активным (зарегистрированным) на данный момент клиентам.

FTP сервер. Реализует “скачивание” и “закачивание” файлов по запросам клиентов.

Распределенный сервер. Несколько серверов “коллективно” обрабатывают запросы клиентов, распределяя части работы между собой. Например,запрос на вычисления разбивается на части, а потом собирается в единый ответ.

Сетевая "игра". Сервер хранит состояние игрового объекта, а клиенты посылают запросы на изменение этого состояния. Например, "морской бой "крестики-нолики"и т.п.

Можно придумать и другие сценарии взаимодействия.

Варианты баз данных (основное задание)

Эти примеры описывают самые простейшие и примитивные данные о реальных событиях или фактах. Можно (и желательно) расширить хранимую информацию, чтобы сделать работу с такой базой более интересной и правдоподобной. При выборе и разработки вашей базы, в первую очередь, нужно понять (придумать) сценарии работы с ней, чтобы эти сценарии описывали действительно нетривиальные процедуры обработки информации, которые могут быть полезны в реальной практике.

---

(1) **Студент 1.**

Запись о студенте содержит ФИО, номер группы, рейтинг, дополнительную информацию (текст)

Запросы к базе. Выборка по всем полям соответственно их назначению, добавление, удаление, редактирование записей.

---

(2) **Студент 2.**

Запись о студенте содержит ФИО, номер группы, и массив отметок экзаменов.

Запросы к базе. Выборка по всем полям соответственно их назначению, добавление, удаление, редактирование оценок. Вычисление средних баллов по группам экзаменов, выборка по этим оценкам.

---

(3) **Ежедневник.**

Запись в ежедневнике состоит из полей: дата, время, событие, длительность

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. Определение свободного времени в данном диапазоне.

---

(4) **Библиотека 1.**

Запись в каталоге библиотеки журналов состоит из полей:
название журнала, год, номер, авторы, название статьи, ключевые слова.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование.

---

(5) **Библиотека 2.**

Запись в каталоге библиотеки книг состоит из полей:
автор, название, жанр, издательство, год выпуска

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование.

---

(6) **Издательство.**

Запись в базе издательства состоит из полей:
автор, название, издательство, год выпуска, цена, тираж, сколько продано

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. Редактирование для прошлых и будущих лет имеет естественные ограничения по смыслу. Также запросы на суммарные тиражи и суммы.

---

(7) **Биллинг 1.**

Запись в базе данных оплаты мобильной связи состоит из полей:
телефон, дата, время, сервис, стоимость

Сервис — это вид обслуживани: разговор, СМС, интернет трафик, и т.д.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. И также выдыча суммарных показателей по стоимости или интернет трафику.

---

(8) **Биллинг 2.**

Запись в базе данных оплаты мобильной связи состоит из полей:
телефон, телефон2, сервис, входящий/исходящий, дата, время, стоимость

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. И также выдыча суммарных показателей по стоимости.

---

(9) **Интернет провайдер.**

Запись в базе данных провайдера состоит из полей:

ФИО абонента, IP адрес, дата, трафик вх/исх за каждый час (24 часа в сутках)

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. И также начисление счетов конкретной выборке за указанный период в соответствии с имеющися почасовым тарифным планом.

---

(10) **СМИ.**

Запись в базе электронного СМИ содержит

дата, время, источник (автор), строка ключевых слов, текст информационной статьи

Запросы к базе. Выборка по всем полям, кроме текста статьи, добавление, удаление, редактирование. Выборка по ключевым словам должна поддерживать простейшую логику (все слова присутствуют, хотя бы одно, не присутствует и т.п.)

---

(11) **Банк .**

Запись в базе банка содержит записи из полей:

номер счета, сумма на счете, тип валюты, сумма операции, дата и время операции, комментарий (кому или от кого идут деньги)

Кроме этого банк имеет таблицу коэффициентов для конвертации валют.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование — выполнение операции зачисления или списания денег в указанной (другой) валюте.

---

(12) **Магазин.**

Запись в базе интернет-магазина содержит записи из полей:

идентификатор покупателя, название товара, количество, цена покупки, дата.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование — выполнение покупки. Также выдача суммарных значений по цене для выборки.

---

(13) **Транспорт 1.**

База по авиаперевозкам между городами. Есть список городов с аэропортами. Есть список рейсов на некоторый период времени между парами городов. Каждая запись о рейсе включает:

Город отправления, город назначения, дата и время вылета, количество свободных и занятых мест, цена билета, стоимость полета и т.п. (тип самолета, авиакомпания ...).

Записи на прошедшие даты считаются фиксированными, записи на будущие даты можно изменять.

Запросы к базе. По типу системы продажи билетов. Для анализа пассажиропотока за период времени. Для анализа затрат и доходов по перевозке в определенных направлениях и по периодам времени.

Моделирующие клиенты производят покупку и сдачу билетов на разные рейсы, т.е. выбирают рейс и занимают часть свободных мест.

---

(14) **Транспорт 2.**

База железнодорожного сообщения. Есть список маршрутов поездов — последовательный список городов (стоимость и время перегонов). Есть расписание рейсов поездов по маршрутам туда и обратно.

рейс, дата и время отправления, время переезда между городами, стоимость переезда между городами, цена билета, количество свободных и занятых мест по перегонам между городами, и т.д.

Записи на прошедшие даты считаются фиксированными, записи на будущие даты можно изменять.

Запросы к базе. По типу системы продажи билетов. Для анализа пассажиропотока за период времени. Для анализа затрат и доходов по перевозке в определенных направлениях и по периодам времени.

Моделирующие клиенты производят покупку и сдачу билетов на разные поездки, т.е. выбирают рейс и занимают часть свободных мест.

---

(15) **ГИБДД.**

База штрафов ГИБДД.

Владелец ФИО, марка машины, гос. номер, штраф: нарушение, дата выдачи, сумма, дата погашения

Запросы к базе. Выдать штраф, погасить штраф. Статистика по наличию, количеству и сумме штрафов по автовладельцам, регионам, отрезкам времени, маркам машин и т.п.

---

(16) **Производственный склад.**

На складе хранятся изделия, которые могут быть изготовлены из других изделий.

Каждая запись базы содержит

наименование изделия, количество, карта сборки

Карта сборки — это названия и количества других изделий, из которых может быть собрано данное. Карта сборки может быть и пустой, т.е. изделие здесь не собирается, а только хранится.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. Кроме этого запрос на сборку — собрать некоторое количество изделий из компонент, хранящихся на этом складе, т.е. удалить компоненты и добавить собранные изделия. Если сборка невозможна, то выдать чего и в каком количестве не хватает.

---

(17) **Почта России.**

Есть список городов, между которыми передаются почтовые отправления. Запись в базе содержит

тип отправления, вес, город отправления, город получения, дата отправления, дата получения

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. Кроме этого суммарные значени по весу для полученной выборки. Тип отправления — это письмо, бандероль, посылка и т.п. Дата получения может быть пустой, т.е. отправление находится в пути.

---

(18) **Миграционная служба.**

База данных о посещении нашей страны мигрантами из других стран. Запись содержит

ФИО, страна происхождения, список дат въезда и выезда, особые отметки

Последняя дата выезда может быть не определена, т.е. человек находится в стране.

Запросы к базе. Выборка по всем полям, добавление, удаление, редактирование. Кроме этого запросы на соблюдение миграционных правил — продолжительность пребывания за год, частота въездов/выездов и т.п. Статистика сколько народу въехало из данной страны за данный период и т.п.

Можно предлагать свои базы под любой разумный набор информации.