1. Класс "Очередь строк".

Реализовать кольцевую очередь строк std::string (длина очереди задаётся аргументом конструктора). Необходимые методы: создание очереди (конструктор), удаление очереди (деструктор), копирование очереди (конструктор копирования и оператор присваивания), перемещение очереди (конструктор перемещения и оператор присваивания перемещением), добавление строки в очередь, удаление строки из очереди, итератор по очереди.

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать создание очереди строк из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

2. Класс "Список строк с возможностью сортировки".

Реализовать однонаправленный список строк std::string. Необходимые методы: инициализация списка (конструктор), удаление списка (деструктор), добавление строки в список, удаление строки из списка, копирование списка, итератор по списку (возможность последовательного перебора элементов списка, начиная с заданного, не использовать оператор []), сортировка списка «пузырьком» в алфавитном порядке. В сортировке не копировать данные одного узла в другой. Реализация списка – классическая ссылочная, с фиктивным элементом.

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать создание списка строк из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

3. Класс "Динамический возрастающий массив чисел".

Требуется реализовать динамический массив целых чисел с возможностью быстрого (бинарного) поиска. Идея реализации состоит в том, что сначала для хранения выделяется небольшой массив фиксированной длины, а по мере добавления элементов выделяются дополнительные такие же массивы, которые связываются в список. Реализация списка – классическая ссылочная.

Необходимые методы: создать массив заданной начальной длины (конструктор), вставить элемент на найденное бинарным поиском место с удлинением массива, удалить элемент по указанному индексу, получить количество элементов в массиве, искать заданное значение в массиве, итератор (возможность последовательного перебора элементов массива, начиная с заданного, не использовать оператор []); удалить все элементы из массива (деструктор), копирование массива (конструктор копирования и оператор присваивания), перемещение массива (конструктор перемещения и оператор присваивания перемещением).

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать заполнение массива элементы из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

4. Класс "Токенайзер текстовой строки".

Требуется разбить данную строку на токены (слова) по заданному набору символов-разделителей и обеспечить работу с этими токенами. Наряду с исходной строкой в реализации создается список, хранящий позиции и длины выделенных токенов. Реализация списка – классическая ссылочная. Строка понимается в стиле Си, длина ее не ограничена.

Необходимые методы: инициализировать токенайзер данным набором символов-разделителей с разбиением строки на токены (конструктор), получить общее количество найденных токенов, итератор токенов (возможность последовательного просмотра, начиная с заданного), получение токена с заданным порядковым номером, разбиение данной строки по другому набору разделителей, копирование токенайзера (конструктор копирования и оператор присваивания), перемещение токенайзера (конструктор перемещения и оператор присваивания перемещением), деструктор.

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать разбиение различных строк по различным разделителям и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

5. Класс "Кольцевая очередь строк".

Требуется реализовать работу очереди строк(в стиле Си) на базе одного отрезка памяти, причем при полном заполнении этого отрезка новые добавленные строки затирают (целиком) строки из начала очереди. Все строки, включая нулевой байт, размещаются в кольцевом буфере памяти без промежутков (при этом одна из строк может оказаться физически разорванной).

Необходимые методы: создать пустую очередь на заданное количество байт (конструктор), добавить строку (в конец очереди, при этом, если произошло затирание, смещается голова очереди), получить длину строки в начале очереди, копировать строку из начала очереди по заданному указателю, удалить начало очереди, получить количество строк в очереди, удалить все элементы очереди (деструктор), получить количество потерянных строк в начале очереди, итератор по очереди (возможность перебора строк, хранящихся в очереди целиком, от ее головы до хвоста), копирование очереди (конструктор копирования и оператор присваивания), перемещение очереди (конструктор перемещения и оператор присваивания перемещением)

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать заполнение очереди строками из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

6. Класс "Скользящее приемное окно".

Пусть имеется набор строк (в стиле Си), занумерованных различными натуральными числами. Эти строки вместе со своими номерами случайным образом подаются на хранение в класс, похожий на очередь, пытающийся разместить строки по порядку их номеров. Можно забирать только группы строк с последовательными номерами. Например, если на хранение поступили строки с номерами 6, 1,3,5,2, то можно забрать только строки с номерами 1, 2, 3, а остальные строки (которые временно сохраняются в другой очереди) можно будет забрать только после поступления строки с номером 4.

Необходимые методы: создать пустую очередь (конструктор кольцевой очереди указателей на char заданной глубины), добавить на хранение в очередь строку с указанным номером (если номер строки не позволяет этого, сохраняем ее в дополнительной очереди, если позволяет, все подходящие строки из дополнительной очереди также добавляются в основную), взять (копировать и удалить) строку из начала (дополнительной) очереди, получить порядковый номер строки в начале очереди, получить максимальный номер строки в очереди, итератор (переборе строк, сохраненные в очереди, от головы до хвоста), удалить все элементы из очереди (деструктор), копирование очереди (конструктор копирования и оператор присваивания), перемещение очереди (конструктор перемещения и оператор присваивания перемещением).

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать заполнение очереди строками, номера которых записаны в первой позиции, из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

7. Класс "Список отрезков".

Реализовать двунаправленный список непересекающихся отрезков на прямой. Необходимые методы: инициализация списка (конструктор), удаление списка (деструктор), копирование списка, итератор по списку (возможность последовательного перебора элементов списка, начиная с заданного, в две стороны), добавление отрезка в список (в этом случае либо добавляется новый отрезок, либо добавляемый отрезок объединяется с имеющимся в списке отрезком, либо отрезок объединяет два или более отрезка в один), удаление отрезка из списка (в этом случае либо со списком ничего не происходит, либо удаляется один или более отрезков списка и (или) обрезается один или два отрезка списка, либо один из отрезков списка делится на две части). Реализация списка – классическая ссылочная, с использованием «умных» указателей.

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать создание списка отрезков из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.

8. Класс "Стек строк с файловым буфером".

Требуется реализовать работу со строками произвольной длины по стековому принципу с ограниченным использованием оперативной памяти. Идея реализации: выделяется блок памяти и строки (длиной не более выделенного блока) укладываются в него одна за другой. Если для добавления новой строки не хватает места, то заполненный блок записывается в файл, а освободившееся место используется для новых добавлений. Если при извлечении строки (удалении вершины) стек окажется пустым, то он заполняется чтением ранее сохраненной в файле информацией (последним записанным туда блоком). Таким образом, в реализации есть два стека: стек строк в памяти и стек блоков записей в файле.

Необходимые методы: создать пустой стек (конструктор), добавить строку, получить длину строки на вершине стека, копировать строку на вершине стека по заданному указателю, удалить вершину стека, получить количество строк в стеке, удалить все элементы стека (деструктор), сохранить состояние стека в файле /загрузить из файла, копирование стека (конструктор копирования и оператор присваивания), перемещение стека (конструктор перемещения и оператор присваивания перемещением), итератор по стеку строк.

Формальное определение интерфейса не задано и должно быть разработано студентом. Тест должен содержать заполнение стека строками из файла и проверку работы всех реализованных методов в различных корректных и некорректных ситуациях.