Лабораторная работа 4.

Контейнеры. Бинарный поиск. Алгоритмы своими руками.

Часть 1. (Максимум 6 баллов)

Выполните следующие задачи по ссылке:

https://informatics.msk.ru/mod/statements/view.php?id=81558#1

Часть 2.

Задача 1. Общие буквы (1 балл)

Вам даны слова. Выведите в алфавитном порядке список общих букв всех слов.

Формат ввода

На вход поступают слова (по одному в строке), состоящие из маленьких латинских букв алфавита. Длина слов не превосходит 100 символов, а количество слов не превосходит 1000.

Формат вывода

Выведите в алфавитном порядке без пробелов список букв, присутствующих в каждом слове.

Примеры.

ВВОД	ВЫВОД
apple peach	aep
alpha beta gamma	A
alpha beta gamma delta epsilon	

Задача 2. (1 балл)

У каждого спортсмена на футболке написан уникальный номер. Спортсмены по очереди выходят из раздевалки и должны построиться на стадионе. Тренер каждому выходящему спортсмену называет номер. Выходящий спортсмен должен встать после спортсмена с таким номером на футболке (а если такого спортсмена на поле нет — встать в начало).

Формат ввода

Сначала задано натуральное число *n*, не превосходящее 100000 — количество спортсменов. Далее идут *n* пар неотрицательных целых чисел, не превосходящих 100000. Первое число в паре — номер очередного выходящего спортсмена. Второе число в паре — номер того спортсмена, после которого текущий должен встать.

Формат вывода

Напечатайте номера спорстменов после построения.

Замечание. Решение с линейным поиском приниматься не будут. Используйте вспомогательную структуру данных, чтобы быстро находить то место, в которое нужно встать.

Примеры.

ВВОД	ВЫВОД
5 42 0 17 42 13 0 123 42 5 13	13 5 42 123 17

Задача 3. Очередь с защитой от ошибок (1 балл)

Научитесь пользоваться стандартной структурой данных queue для целых чисел. Напишите программу, содержащую описание очереди и моделирующую работу очереди, реализовав все указанные здесь методы.

Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку.

Возможные команды для программы:

push	Добавить в очередь число n (значение n задается после команды). Программа должна вывести ok.
pop	Удалить из очереди первый элемент. Программа должна вывести его значение.
front	Программа должна вывести значение первого элемента, не удаляя его из очереди.
size	Программа должна вывести количество элементов в очереди.
clear	Программа должна очистить очередь и вывести ok.
exit	Программа должна вывести bye и завершить работу.

Перед исполнением операций front и pop программа должна проверять, содержится ли в очереди хотя бы один элемент. Если во входных данных встречается операция front или pop, и при этом очередь пуста, то программа должна вместо числового значения вывести строку error.

Формат ввода

Вводятся команды управления очередью, по одной на строке

Формат вывода

Требуется вывести протокол работы очереди, по одному сообщению на строке

Примеры.

ВВОД	ВЫВОД
push 1 front exit	ok 1 Bye
size push 1 size push 2 size push 3 size exit	0 ok 1 ok 2 ok 3 bye
push 3 push 14 size clear push 1 front push 2 front pop size pop size exit	ok ok 2 ok ok 1 ok 1 1 1 2 0 bye

Задача 4. Очередь с приоритетами (2 балла)

Напишите класс, который будет обрабатывать последовательность запросов таких видов:

CLEAR — сделать очередь с приоритетами пустой (если в очереди уже были какие-то элементы, удалить все). Действие происходит только с данными в памяти, на экран ничего не выводится.

ADD n — добавить в очередь с приоритетами число n. Действие происходит только с данными в памяти, на экран ничего не выводится.

EXTRACT — вынуть из очереди с приоритетами максимальное значение. Следует и изменить данные в памяти, и вывести на экран или найденное максимальное значение, или, если очередь была пустой, слово "CANNOT" (большими буквами).

Формат ввода

Во входных данных записано произвольную последовательность запросов CLEAR, ADD и EXTRACT — каждый в отдельной строке, согласно вышеописанному формату.Суммарное количество всех запросов не превышает 200000.

Формат вывода

Для каждого запроса типа EXTRACT выведите на стандартный выход (экран) его результат (в отдельной строке).

Замечание.

Следует использовать стандартную реализацию очереди с приоритетами в STL; она называется priority_queue, для её использования необходимо подключить заголовочный файл queue.

Примеры.

ВВОД	ВЫВОД
ADD 192168812 ADD 125 ADD 321 EXTRACT EXTRACT CLEAR ADD 7 ADD 555 EXTRACT EXTRACT EXTRACT	192168812 321 555 7 CANNOT
CLEAR ADD 7 ADD 555 EXTRACT EXTRACT EXTRACT CLEAR EXTRACT ADD 321847 EXTRACT EXTRACT ADD 127307 ADD 949912	555 7 CANNOT CANNOT 321847 CANNOT 949912 840884

ADD 840884 ADD 654060 EXTRACT EXTRACT	
ADD 281305 ADD 876406 ADD 960204 ADD 562475 EXTRACT EXTRACT ADD 1035495 EXTRACT EXTRACT EXTRACT	960204 876406 1035495 562475 281305

Далее во всех задачах подразумевается что вход шаблонных функций - поступают итераторы.

Задача 5. Any_of (2 балла)

Вам надо написать свою реализацию стандартного алгоритма **any_of**. Заголовок функции должен быть таким:

template <typename It, typename Pred> bool any_of(It first, It last, Pred f);

Функция должна вернуть true, если в последовательности [first; last) существует элемент, для которого функция f возвращает истину. В противном случае функция должна вернуть false.

Задача 6. remove_copy_if (2 балла)

Вам надо написать свою реализацию стандартного алгоритма **remove_copy_if**. Заголовок функции должен быть таким:

template <typename InIter, typename OutIter, typename Predicate> OutIter remove_copy_if(InIter first, InIter last, OutIter out, Predicate f);

Функция должна копировать из подпоследовательности [first; last) в последовательность, начинающуюся с out, те элементы, которые не удовлетворяют предикату f. Функция возвращает итератор, указывающий за последний скопированный элемент.

Задача 7. set_difference (2 балла)

Вам надо написать свою реализацию стандартного алгоритма set_difference. Заголовок функции должен быть таким:

template <typename InIter1, typename InIter2, typename OutIter>

OutIter set_difference(InIter1 first1, InIter1 last1, InIter2 first2, InIter2 last2, OutIter out);

Под словом set (множество) здесь понимается не математическое множество и даже не контейнер std::set, а просто отсортированная последовательность.

Функция должна сформировать элементы разности множеств [first1, last1) и [first2, last2) и записать их в последовательность, начинающуюся с out. Исходные интервалы предполагаются отсортированными. Каждый элемент считается со своей кратностью. Функция возвращает итератор, указывающий за последний записанный элемент.

В этой задаче запрещено пользоваться встроенными контейнерами и функциями, вам надо полностью реализовать свой алгоритм.

Задача 8. (3 балла)

Вам нужно написать функцию sideways_merge, которая сливает две последовательности [first1, last1) и [first2, last2), первая из которых отсортирована по возрастанию, а вторая - по убыванию, в одну отсортированную по возрастанию последовательность.

Функцию оформите следующим образом:

```
template <typename It, typename OutputIt>
OutputIt sideways_merge(It first1, It last1, It first2, It last2, OutputIt out);
```

Функция должна записывать итоговую последовательность начиная с out. Функция возвращает итератор, указывающий за последний скопированный элемент.

В этой задаче запрещено пользоваться встроенными контейнерами и функциями, вам надо полностью реализовать свой алгоритм. Если планируете использовать сортировку, то ее надо реализовать собственными силами, и один из двух вариантов: Быстрая сортировка или Сортировка слиянием.