Организационные вопросы

Оценка лабораторных работ:

Лабораторная работа оценивается по 2-балльной шкале (зачтено/не зачтено). Для получения положительной оценки достаточно выполнить работу с соблюдением минимальных требований. Выполнение дополнительных заданий не отменяет необходимость соблюдения минимальных требований.

Минимальные требования

1. Код должен компилироваться без предупреждений при максимальном уровне предупреждений. Для компилятора GCC это набор флагов: -Wall -Wconversion -Wextra -Wpedantic. Для компилятора MSVC (Visual Studio) это флаг /W4. Для других компиляторов согласуйте настройки компиляции с преподавателем.
2. Нельзя использовать глобальные переменные (константы допустимы).
3. В коде должен использоваться только полноценный английский язык (***транслит запрещен***). Русский язык разрешено использовать только в комментариях.
4. Запрещается комментировать каждую строчку кода. Допустим один краткий комментарий на блок кода. Разрешается комментировать одиночную строчку кода, только если она **действительно** делает что-то неожидаемое и хитрое (но помните, хитрый код — плохой код!).
5. Весь код должен удовлетворять единому стилю программирования. Сам стиль можно выбирать по своему вкусу (см., например, Google C++ Style Guide, GNU C Style и другие). То есть запрещено, например, называть одну функцию в стиле CamelCase, а другую — в стиле snake_case. Исключение допустимо только для названия функции main, которое всегда пишется в нижнем регистре. Данные требования предъявляются к любым именуемым сущностям в программе — к функциям, методам, классам, локальным переменным, параметрам функций и методов, названиям файлов и так далее.
6. В случае удалённого формата сдачи обучающийся предоставляет ссылку на git-репозиторий. В случае очного формата сдачи репозиторий может быть локальным (на компьютере в аудитории или ноутбуке обучающегося). В любом случае репозиторий не может содержать одинокий коммит с целой лабой, а должен показывать историю работы над лабораторной в виде серии коммитов с содержательными заголовками. В репозитории должен быть корректным образом настроен файл .gitignore (т. е. в репозитории должны находиться только файлы с исходным кодом и файлами проекта, никаких промежуточных и итоговых результатов компиляции в нём быть не должно).
7. В случае удалённого формата сдачи для защиты необходимо предоставить небольшой содержательный отчёт, подготовленный в Latex, LibreOffice, Microsoft Word или любой другой системе. Отчёт предоставляется в формате pdf (по запросу преподавателя - в исходном формате). В отчёте должны присутствовать: титульный лист, выданное согласно варианту задание, протокол тестирования, заключение по выполненной работе и дополнительные разделы, требуемые вашим преподавателем (если есть).

Лабораторная работа № 1: Деревья поиска

Лабораторная работа состоит из 2 частей:
- написания класса согласно варианту;
- решения одной из практических задач с помощью написанного класса.

Решением задачи является функция вне класса (не метод класса и не friend-функция).

Варианты и задачи распределяются преподавателем.

Общие требования

Реализовать аналог класса std::set<int>.

Класс должен представлять собой двоичное дерево поиска.
Стандартные контейнеры в качестве полей класса не использовать.

Класс должен предоставлять, как минимум, следующие функции:

1. конструктор копирования;
2. деструктор;
3. оператор присваивания;
4. void print() – печать содержимого;
5. bool insert(int key) – вставка элемента;
6. bool contains(int key) - проверка наличия элемента;
7. bool erase(int key) – удаление элемента;

Для написанного контейнера, получите:
Среднее время заполнения 1000, 10000 и 100000 уникальными случайными числами (среднее рассчитывать по 100 попыткам, функцию-генератор можно взять из приложения);
Среднее время поиска случайного числа в заполненном контейнере из 1000, 10000 и 100000 элементов (среднее рассчитывать по 1000 попыткам);
Среднее время добавления и удаления случайного числа для заполненного контейнера из 1000, 10000 и 100000 элементов (среднее рассчитывать по 1000 попыткам);

Сравните полученное время с теми же значениями для класса std::vector<int>.

Эксперименты допускается проводить до сдачи лабораторной (код для проведения экспериментов и полученные данные должен быть в наличии).

Задачи:

Вариант 1: напишите функции, возвращающие пересечение и разность 2-х множеств целых чисел.

Вариант 2: напишите функции, возвращающие объединение и симметрическую разность 2-х множеств целых чисел.

Вариант 3: для заданного std::vector<int> верните новый std::vector<int>, содержащий все повторяющиеся элементы (для вектора {3 2 2 4} результат должен быть {2})

Вариант 4: для заданного std::vector<int> верните новый std::vector<int>, содержащий все неповторяющиеся элементы (для вектора {3 2 2 4} результат должен быть {3 4})

Доп. задания (по желанию студента):

Создать копию вашего класса. Изменить копию так, чтобы полученный класс допускал хранение дублирующихся ключей. В полученном классе реализовать функцию size_t count(int x), возвращающую количество элементов, равных x.

Реализовать шаблонный класс (аналог std::set<T>) вместо обычного класса;

Реализовать балансировку дерева;

Реализовать методы begin() и end(), возвращающие итераторы, позволяющие обходить дерево (порядок обхода может быть любым).

Приложение А. Функция-генератор псевдослучайных числел

```
size_t lcg(){
    static size_t x = 0;
    x = (1021*x+24631) % 116640;
    return x;
}
```