Лабораторная работа № 3 по курсу дискретного анализа: Исследование качества программ

Выполнил студент группы М8О-209Б-23 *Кривошапкин Егор*.

# Условие

Кратко описывается задача:

1. Для реализации словаря из предыдущей лабораторной работы необходимо провести исследование скорости выполнения и потребления оперативной памяти.

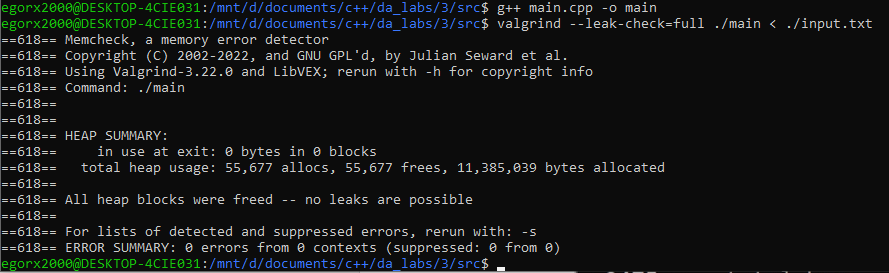
# Метод решения

Изучение утилит для исследования качества программ таких как gcov, gprof, valgrind, и их использование для оптимизации программы.

# Valgrind

Valgrind — инструментальное программное обеспечение, предназначенное для отладки использования памяти, обнаружения утечек памяти, а также профилирования.

В ходе выполнения лабораторной работы утилита будет использована исключительно для поиска утечек и отладки использования памяти.

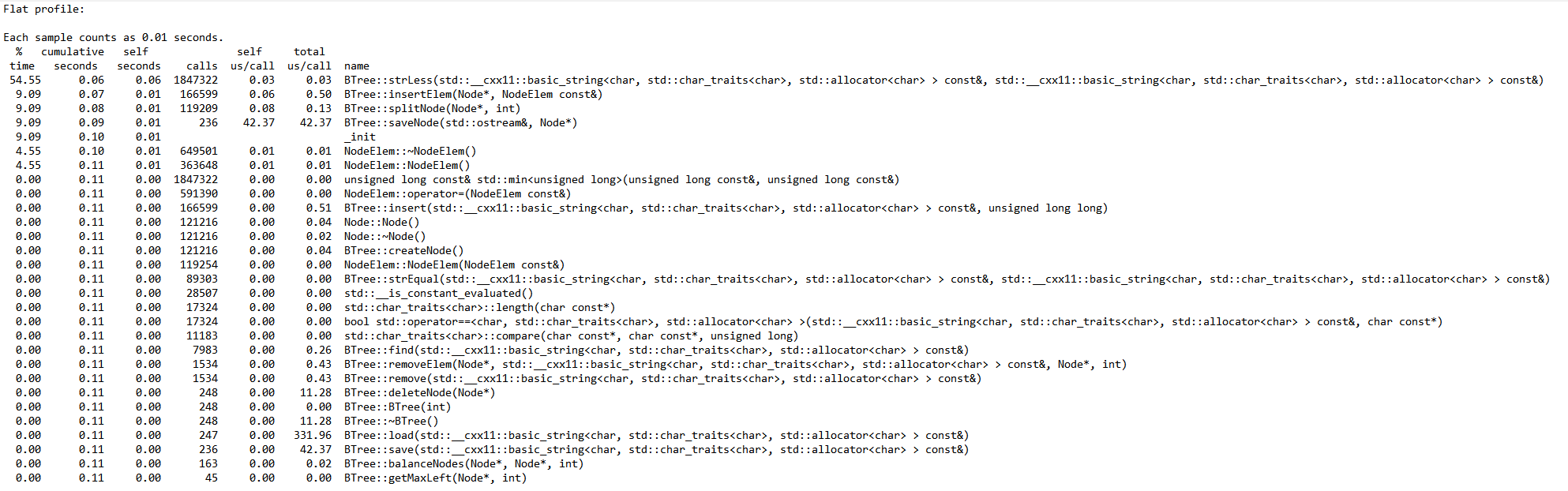


Как видим, утечек памяти с помощью Valgrind обнаружено не было, и моя программа корректно управляет памятью. При использовании моего теста было произведено 55677 аллокаций и освобождений памяти.

# gprof

Gprof - это инструмент для профилирования программы. Благодаря нему мы можем отследить, где и сколько времени проводила программа, тем самым выявляя слабые участки.

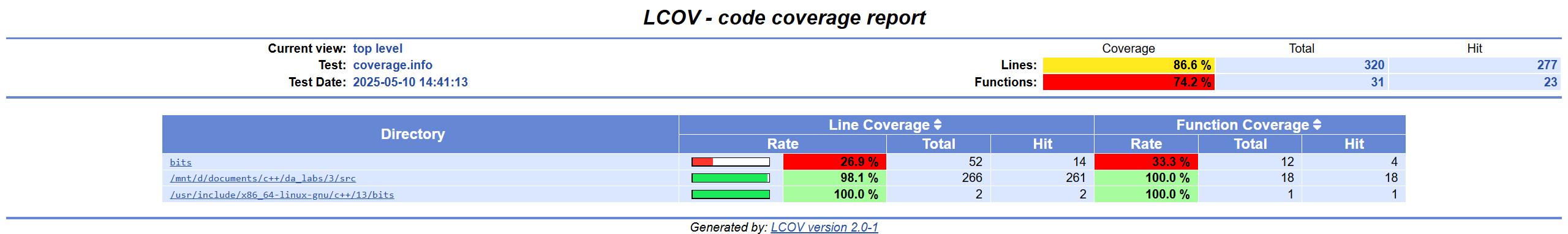
Применю gprof на том же тесте.

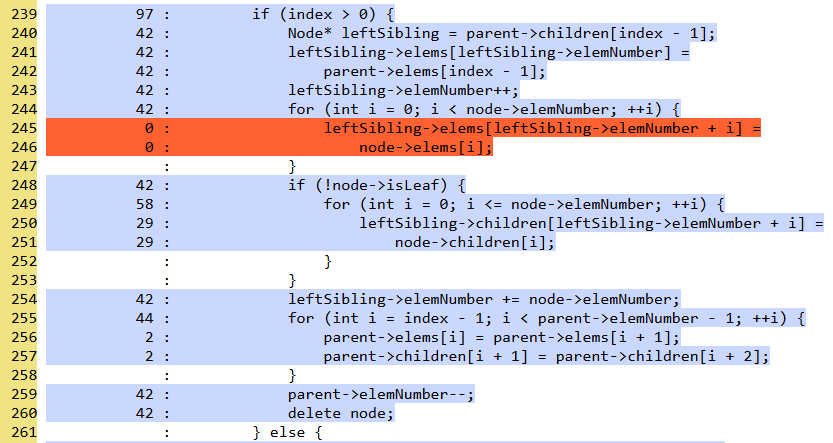
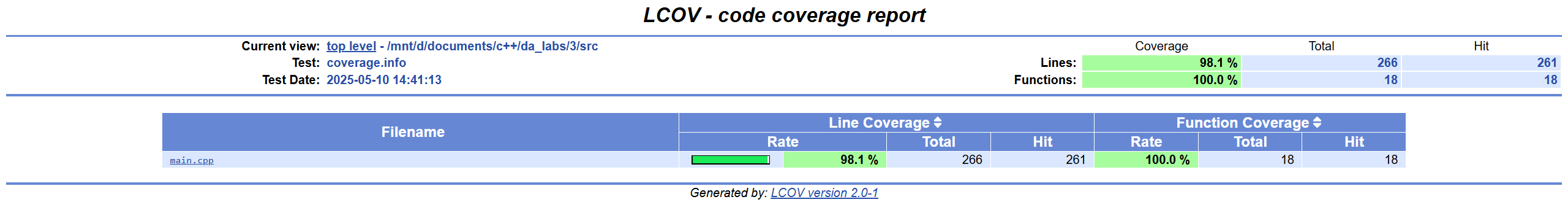


Как видим, большая часть времени работы программы (54,55%) тратится на сравнение строк, поскольку в функции используется посимвольное сравнение в цикле без оптимизаций, к тому же для каждого символа вызывается нетривиальная операция понижения регистра.

# gcov

Gcov — свободно распространяемая утилита для исследования покрытия кода. Gcov генерирует точное количество исполнений для каждого оператора в программе и позволяет добавить аннотации к исходному коду. С помощью утилит lcov и genhtml можно получить html страницу с отчетом покрытия кода. Проверим покрытия кода при выполнении моей программой всё того же теста, получив ту самую html-страницу:





Как видим, в непосредственно самой реализации B-Дерева покрыто 98*.*1% кода. Это объясняется тем, что в тесте исследуются не все случаи работы с деревом. Например, при удалении ни разу не произошло слияния узла с левым соседом.

# Выводы

Я познакомился с инструментами, позволяющими находить недостатки даже в, казалось бы, полностью рабочих программах:

1. valgrind позволяет выявлять утечки памяти и профилировать код.
2. gprof позволяет оценить производительность программы, посмотреть, какие структуры и функции замедляют программу, выявить слабые места в производительности.
3. gcov позволяет исследовать покрытие кода, определив его участки, которые можно удалить или сократить. Можно сгенерировать страницу формата html, в которой будет показано наше покрытие.

Инструменты являются крайне полезными. В дальнейшем будущем я буду стараться использовать их как можно чаще, что, несомненно, повысит качество написанных мною программ.