Московский авиационный институт
(Национальный исследовательский университет)
Факультет "Информационные технологии и прикладная математика"
Кафедра "Вычислительная математика и программирование"

Лабораторная работа №3 по курсу “Дискретный анализ”

Студент: Иванов Андрей Кириллович

Группа: М8О-208Б-22

Преподаватель: Макаров Н.К.

Вариант: 0

Оценка: _____

Дата: _____

Подпись: _____

Москва, 2024

Содержание

1 Задание

Для реализации словаря из предыдущей лабораторной работы, необходимо провести исследование скорости выполнения и потребления оперативной памяти. В случае выявления ошибок или явных недочётов, требуется их исправить.

2 Метод решения

Результатом лабораторной работы является отчёт, состоящий из:

Дневника выполнения работы, в котором отражено что и когда делалось, какие средства использовались и какие результаты были достигнуты на каждом шаге выполнения лабораторной работы. Выводов о найденных недочётах. Сравнение работы исправленной программы с предыдущей версией. Общих выводов о выполнении лабораторной работы, полученном опыте.

Минимальный набор используемых средств должен содержать утилиту gprof и библиотеку dmalloc, однако их можно заменять на любые другие аналогичные или более развитые утилиты (например, Valgrind или Shark) или добавлять к ним новые (например, gcov).

3 Valgrind

valgrind

```
roman@roman-BMH-WDX9:~/Discr_Labs/2$ valgrind ./test
==7204== Memcheck, a memory error detector
==7204== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==7204== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==7204== Command: ./test
==7204==
==7204==
==7204== HEAP SUMMARY:
==7204==     in use at exit: 0 bytes in 0 blocks
==7204==   total heap usage: 200,001 allocs, 200,001 frees, 34,072,704 bytes allocated
==7204==
==7204== All heap blocks were freed -- no leaks are possible
==7204==
==7204== For lists of detected and suppressed errors, rerun with: -s
==7204== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

4 Gprof

gprof

```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 44.44      0.12     0.12   100000     0.00     0.00  search(node
   *, char*)
 22.22      0.18     0.06   100000     0.00     0.00  genKey(char
   *)
 14.81      0.22     0.04   100000     0.00     0.00  insert(node
   *&, node*)
  7.41      0.24     0.02        1    20.00    20.00  destroy(node
   *)
  3.70      0.25     0.01   100000     0.00     0.00  std::chrono
   ::duration<double, std::ratio<1l, 1000l> > std::chrono::
   __duration_cast_impl<std::chrono::duration<double, std::ratio<1
   l, 1000l> >, std::ratio<1l, 10000000l>, double, true, false>::
   __cast<long, std::ratio<1l, 10000000000l> >(std::chrono::
   duration<long, std::ratio<1l, 10000000000l> > const&)
  3.70      0.26     0.01                             _init
  3.70      0.27     0.01                             main
  0.00      0.27     0.00   300000     0.00     0.00  std::chrono
   ::duration<long, std::ratio<1l, 10000000000l> >::count() const
  0.00      0.27     0.00   200000     0.00     0.00  std::chrono
   ::time_point<std::chrono::_V2::system_clock, std::chrono::
   duration<long, std::ratio<1l, 10000000000l> > >::
   time_since_epoch() const
  0.00      0.27     0.00   200000     0.00     0.00  std::chrono
   ::duration<double, std::ratio<1l, 1000l> >::count() const
  0.00      0.27     0.00   100000     0.00     0.00  node::node(
   char*, unsigned long)
  0.00      0.27     0.00   100000     0.00     0.00  node::~node
   ()
  0.00      0.27     0.00   100000     0.00     0.00  std::
   enable_if<std::chrono::__is_duration<std::chrono::duration<
   double, std::ratio<1l, 1000l> > >::value, std::chrono::duration
   <double, std::ratio<1l, 1000l> > >::type std::chrono::
   duration_cast<std::chrono::duration<double, std::ratio<1l, 1000
   l> >, long, std::ratio<1l, 10000000000l> >(std::chrono::duration
   <long, std::ratio<1l, 10000000000l> > const&)
  0.00      0.27     0.00   100000     0.00     0.00  std::chrono
   ::duration<double, std::ratio<1l, 1000l> >::duration<long, std
   ::ratio<1l, 10000000000l>, void>(std::chrono::duration<long, std
   ::ratio<1l, 10000000000l> > const&)
  0.00      0.27     0.00   100000     0.00     0.00  std::chrono
   ::duration<double, std::ratio<1l, 1000l> >::duration<double,
   void>(double const&)
  0.00      0.27     0.00   100000     0.00     0.00  std::chrono
   ::duration<long, std::ratio<1l, 10000000000l> >::duration<long,
   void>(long const&)
  0.00      0.27     0.00   100000     0.00     0.00  std::
   common_type<std::chrono::duration<long, std::ratio<1l,
   10000000000l> >, std::chrono::duration<long, std::ratio<1l,
   10000000000l> > >::type std::chrono::operator-<std::chrono::_V2
   ::system_clock, std::chrono::duration<long, std::ratio<1l,
   10000000000l> >, std::chrono::duration<long, std::ratio<1l,
```

```
    10000000001> > >(std::chrono::time_point<std::chrono::_V2::
    system_clock, std::chrono::duration<long, std::ratio<1l,
    10000000001> > > const&, std::chrono::time_point<std::chrono::
    _V2::system_clock, std::chrono::duration<long, std::ratio<1l,
    10000000001> > > const&)
  0.00      0.27     0.00   100000     0.00     0.00  std::
    common_type<std::chrono::duration<long, std::ratio<1l,
    10000000001> >, std::chrono::duration<long, std::ratio<1l,
    10000000001> > >::type std::chrono::operator-<long, std::ratio
    <1l, 10000000001>, long, std::ratio<1l, 10000000001> >(std::
    chrono::duration<long, std::ratio<1l, 10000000001> > const&,
    std::chrono::duration<long, std::ratio<1l, 10000000001> > const
    &)
  0.00      0.27     0.00    66627     0.00     0.00  split(node*,
     node*&, node*&, char*)
  0.00      0.27     0.00        1     0.00     0.00
    __static_initialization_and_destruction_0(int, int)

 %         the percentage of the total running time of the
time       program used by this function.

cumulative a running sum of the number of seconds accounted
 seconds   for by this function and those listed above it.

 self      the number of seconds accounted for by this
seconds    function alone.  This is the major sort for this
           listing.

calls      the number of times this function was invoked, if
           this function is profiled, else blank.

 self      the average number of milliseconds spent in this
ms/call    function per call, if this function is profiled,
     else blank.

 total     the average number of milliseconds spent in this
ms/call    function and its descendents per call, if this
     function is profiled, else blank.

name       the name of the function.  This is the minor sort
           for this listing. The index shows the location of
     the function in the gprof listing. If the index is
     in parenthesis it shows where it would appear in
     the gprof listing if it were to be printed.


Copyright (C) 2012-2022 Free Software Foundation, Inc.

Copying and distribution of this file, with or without
    modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.


         Call graph (explanation follows)


granularity: each sample hit covers 4 byte(s) for 3.70% of 0.27
    seconds

```

```
index % time    self  children    called     name
                                                 <spontaneous>
[1]     96.3    0.01    0.25                 main [1]
                0.12    0.00  100000/100000      search(node*, char*) [2]
                0.06    0.00  100000/100000      genKey(char*) [3]
                0.04    0.00  100000/100000      insert(node*&, node*) [4]
                0.02    0.00       1/1           destroy(node*) [5]
                0.00    0.01  100000/100000      std::chrono::duration<double, std::ratio<1l, 1000l> >::duration<long, std::ratio<1l, 10000000001>, void>(std::chrono::duration<long, std::ratio<1l, 10000000001> > const&) [8]
                0.00    0.00  100000/100000      node::node(char*, unsigned long) [19]
                0.00    0.00  100000/100000      std::common_type<std::chrono::duration<long, std::ratio<1l, 10000000001> >, std::chrono::duration<long, std::ratio<1l, 10000000001> > >::type std::chrono::operator-<std::chrono::_V2::system_clock, std::chrono::duration<long, std::ratio<1l, 10000000001> >, std::chrono::duration<long, std::ratio<1l, 10000000001> > >(std::chrono::time_point<std::chrono::_V2::system_clock, std::chrono::duration<long, std::ratio<1l, 10000000001> > > const&, std::chrono::time_point<std::chrono::_V2::system_clock, std::chrono::duration<long, std::ratio<1l, 10000000001> > > const&) [23]
                0.00    0.00  100000/200000      std::chrono::duration<double, std::ratio<1l, 1000l> >::count() const [18]
-----------------------------------------------
                              1969081             search(node*, char*) [2]
                0.12    0.00  100000/100000      main [1]
[2]     44.4    0.12    0.00  100000+1969081 search(node*, char*) [2]
                              1969081             search(node*, char*) [2]
-----------------------------------------------
                0.06    0.00  100000/100000      main [1]
[3]     22.2    0.06    0.00  100000          genKey(char*) [3]
-----------------------------------------------
                              1769454             insert(node*&, node*) [4]
                0.04    0.00  100000/100000      main [1]
[4]     14.8    0.04    0.00  100000+1769454 insert(node*&, node*) [4]
                0.00    0.00   66627/66627       split(node*, node*&, node*&, char*) [25]
                              1769454             insert(node*&, node*) [4]
-----------------------------------------------
                               200000             destroy(node*) [5]
                0.02    0.00       1/1           main [1]
[5]      7.4    0.02    0.00       1+200000  destroy(node*) [5]
                0.00    0.00  100000/100000      node::~node() [20]
                               200000             destroy(node*) [5]
-----------------------------------------------
```

```
97                0.00    0.01  100000/100000      std::chrono::
    duration<double, std::ratio<1l, 1000l> >::duration<long, std::
    ratio<1l, 10000000001>, void>(std::chrono::duration<long, std::
    ratio<1l, 10000000001> > const&) [8]
98 [6]      3.7    0.00    0.01  100000         std::enable_if<std::
    chrono::__is_duration<std::chrono::duration<double, std::ratio
    <1l, 1000l> > >::value, std::chrono::duration<double, std::
    ratio<1l, 1000l> > >::type std::chrono::duration_cast<std::
    chrono::duration<double, std::ratio<1l, 1000l> >, long, std::
    ratio<1l, 10000000001> >(std::chrono::duration<long, std::ratio
    <1l, 10000000001> > const&) [6]
99                0.01    0.00  100000/100000      std::chrono::
    duration<double, std::ratio<1l, 1000l> > std::chrono::
    __duration_cast_impl<std::chrono::duration<double, std::ratio<1
    l, 1000l> >, std::ratio<1l, 10000001>, double, true, false>::
    __cast<long, std::ratio<1l, 10000000001> >(std::chrono::
    duration<long, std::ratio<1l, 10000000001> > const&) [7]
100 -----------------------------------------------
101               0.01    0.00  100000/100000      std::enable_if<
    std::chrono::__is_duration<std::chrono::duration<double, std::
    ratio<1l, 1000l> > >::value, std::chrono::duration<double, std
    ::ratio<1l, 1000l> > >::type std::chrono::duration_cast<std::
    chrono::duration<double, std::ratio<1l, 1000l> >, long, std::
    ratio<1l, 10000000001> >(std::chrono::duration<long, std::ratio
    <1l, 10000000001> > const&) [6]
102 [7]      3.7    0.01    0.00  100000         std::chrono::duration
    <double, std::ratio<1l, 1000l> > std::chrono::
    __duration_cast_impl<std::chrono::duration<double, std::ratio<1
    l, 1000l> >, std::ratio<1l, 10000001>, double, true, false>::
    __cast<long, std::ratio<1l, 10000000001> >(std::chrono::
    duration<long, std::ratio<1l, 10000000001> > const&) [7]
103               0.00    0.00  100000/300000      std::chrono::
    duration<long, std::ratio<1l, 10000000001> >::count() const
    [16]
104               0.00    0.00  100000/100000      std::chrono::
    duration<double, std::ratio<1l, 1000l> >::duration<double, void
    >(double const&) [21]
105 -----------------------------------------------
106               0.00    0.01  100000/100000      main [1]
107 [8]      3.7    0.00    0.01  100000         std::chrono::duration
    <double, std::ratio<1l, 1000l> >::duration<long, std::ratio<1l,
     10000000001>, void>(std::chrono::duration<long, std::ratio<1l,
     10000000001> > const&) [8]
108               0.00    0.01  100000/100000      std::enable_if<
    std::chrono::__is_duration<std::chrono::duration<double, std::
    ratio<1l, 1000l> > >::value, std::chrono::duration<double, std
    ::ratio<1l, 1000l> > >::type std::chrono::duration_cast<std::
    chrono::duration<double, std::ratio<1l, 1000l> >, long, std::
    ratio<1l, 10000000001> >(std::chrono::duration<long, std::ratio
    <1l, 10000000001> > const&) [6]
109               0.00    0.00  100000/200000      std::chrono::
    duration<double, std::ratio<1l, 1000l> >::count() const [18]
110 -----------------------------------------------
111                                                  <spontaneous>
112 [9]      3.7    0.01    0.00                 _init [9]
113 -----------------------------------------------
114               0.00    0.00  100000/300000      std::chrono::
    duration<double, std::ratio<1l, 1000l> > std::chrono::
    __duration_cast_impl<std::chrono::duration<double, std::ratio<1
```

```
    l, 1000l> >, std::ratio<1l, 10000000l>, double, true, false>::
    __cast<long, std::ratio<1l, 10000000000l> >(std::chrono::
    duration<long, std::ratio<1l, 10000000000l> > const&) [7]
115                 0.00    0.00  200000/300000      std::common_type<
    std::chrono::duration<long, std::ratio<1l, 10000000000l> >, std
    ::chrono::duration<long, std::ratio<1l, 10000000000l> > >::type
    std::chrono::operator-<long, std::ratio<1l, 10000000000l>, long,
     std::ratio<1l, 10000000000l> >(std::chrono::duration<long, std
    ::ratio<1l, 10000000000l> > const&, std::chrono::duration<long,
    std::ratio<1l, 10000000000l> > const&) [24]
116 [16]     0.0    0.00    0.00  300000         std::chrono::duration
    <long, std::ratio<1l, 10000000000l> >::count() const [16]
117 -----------------------------------------------
118                 0.00    0.00  200000/200000      std::common_type<
    std::chrono::duration<long, std::ratio<1l, 10000000000l> >, std
    ::chrono::duration<long, std::ratio<1l, 10000000000l> > >::type
    std::chrono::operator-<std::chrono::_V2::system_clock, std::
    chrono::duration<long, std::ratio<1l, 10000000000l> >, std::
    chrono::duration<long, std::ratio<1l, 10000000000l> > >(std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000000l> > > const&, std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000000l> > > const&) [23]
119 [17]     0.0    0.00    0.00  200000         std::chrono::
    time_point<std::chrono::_V2::system_clock, std::chrono::
    duration<long, std::ratio<1l, 10000000000l> > >::
    time_since_epoch() const [17]
120 -----------------------------------------------
121                 0.00    0.00  100000/200000      main [1]
122                 0.00    0.00  100000/200000      std::chrono::
    duration<double, std::ratio<1l, 1000l> >::duration<long, std::
    ratio<1l, 10000000000l>, void>(std::chrono::duration<long, std::
    ratio<1l, 10000000000l> > const&) [8]
123 [18]     0.0    0.00    0.00  200000         std::chrono::duration
    <double, std::ratio<1l, 1000l> >::count() const [18]
124 -----------------------------------------------
125                 0.00    0.00  100000/100000      main [1]
126 [19]     0.0    0.00    0.00  100000         node::node(char*,
    unsigned long) [19]
127 -----------------------------------------------
128                 0.00    0.00  100000/100000      destroy(node*)
    [5]
129 [20]     0.0    0.00    0.00  100000         node::~node() [20]
130 -----------------------------------------------
131                 0.00    0.00  100000/100000      std::chrono::
    duration<double, std::ratio<1l, 1000l> > std::chrono::
    __duration_cast_impl<std::chrono::duration<double, std::ratio<1
    l, 1000l> >, std::ratio<1l, 10000000l>, double, true, false>::
    __cast<long, std::ratio<1l, 10000000000l> >(std::chrono::
    duration<long, std::ratio<1l, 10000000000l> > const&) [7]
132 [21]     0.0    0.00    0.00  100000         std::chrono::duration
    <double, std::ratio<1l, 1000l> >::duration<double, void>(double
     const&) [21]
133 -----------------------------------------------
134                 0.00    0.00  100000/100000      std::common_type<
    std::chrono::duration<long, std::ratio<1l, 10000000000l> >, std
    ::chrono::duration<long, std::ratio<1l, 10000000000l> > >::type
    std::chrono::operator-<long, std::ratio<1l, 10000000000l>, long,
     std::ratio<1l, 10000000000l> >(std::chrono::duration<long, std
```

```
    ::ratio<1l, 10000000001> > const&, std::chrono::duration<long,
    std::ratio<1l, 10000000001> > const&) [24]
[22]     0.0    0.00    0.00  100000         std::chrono::duration
    <long, std::ratio<1l, 10000000001> >::duration<long, void>(long
     const&) [22]
-----------------------------------------------
                0.00    0.00  100000/100000      main [1]
[23]     0.0    0.00    0.00  100000         std::common_type<std
    ::chrono::duration<long, std::ratio<1l, 10000000001> >, std::
    chrono::duration<long, std::ratio<1l, 10000000001> > >::type
    std::chrono::operator-<std::chrono::_V2::system_clock, std::
    chrono::duration<long, std::ratio<1l, 10000000001> >, std::
    chrono::duration<long, std::ratio<1l, 10000000001> > >(std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000001> > > const&, std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000001> > > const&) [23]
                0.00    0.00  200000/200000      std::chrono::
    time_point<std::chrono::_V2::system_clock, std::chrono::
    duration<long, std::ratio<1l, 10000000001> > >::
    time_since_epoch() const [17]
                0.00    0.00  100000/100000      std::common_type<
    std::chrono::duration<long, std::ratio<1l, 10000000001> >, std
    ::chrono::duration<long, std::ratio<1l, 10000000001> > >::type
    std::chrono::operator-<long, std::ratio<1l, 10000000001>, long,
     std::ratio<1l, 10000000001> >(std::chrono::duration<long, std
    ::ratio<1l, 10000000001> > const&, std::chrono::duration<long,
    std::ratio<1l, 10000000001> > const&) [24]
-----------------------------------------------
                0.00    0.00  100000/100000      std::common_type<
    std::chrono::duration<long, std::ratio<1l, 10000000001> >, std
    ::chrono::duration<long, std::ratio<1l, 10000000001> > >::type
    std::chrono::operator-<std::chrono::_V2::system_clock, std::
    chrono::duration<long, std::ratio<1l, 10000000001> >, std::
    chrono::duration<long, std::ratio<1l, 10000000001> > >(std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000001> > > const&, std::
    chrono::time_point<std::chrono::_V2::system_clock, std::chrono
    ::duration<long, std::ratio<1l, 10000000001> > > const&) [23]
[24]     0.0    0.00    0.00  100000         std::common_type<std
    ::chrono::duration<long, std::ratio<1l, 10000000001> >, std::
    chrono::duration<long, std::ratio<1l, 10000000001> > >::type
    std::chrono::operator-<long, std::ratio<1l, 10000000001>, long,
     std::ratio<1l, 10000000001> >(std::chrono::duration<long, std
    ::ratio<1l, 10000000001> > const&, std::chrono::duration<long,
    std::ratio<1l, 10000000001> > const&) [24]
                0.00    0.00  200000/300000      std::chrono::
    duration<long, std::ratio<1l, 10000000001> >::count() const
    [16]
                0.00    0.00  100000/100000      std::chrono::
    duration<long, std::ratio<1l, 10000000001> >::duration<long,
    void>(long const&) [22]
-----------------------------------------------
                               199627             split(node*, node
    *&, node*&, char*) [25]
                0.00    0.00   66627/66627       insert(node*&,
    node*) [4]
[25]     0.0    0.00    0.00   66627+199627  split(node*, node*&,
    node*&, char*) [25]
```

```
                                199627             split(node*, node
    *&, node*&, char*) [25]
-----------------------------------------------
                0.00    0.00       1/1
    _GLOBAL__sub_I__Z7destroyP4node [27]
[26]     0.0    0.00    0.00       1
    __static_initialization_and_destruction_0(int, int) [26]
-----------------------------------------------

 This table describes the call tree of the program, and was sorted
     by
 the total amount of time spent in each function and its children.

 Each entry in this table consists of several lines.  The line
    with the
 index number at the left hand margin lists the current function.
 The lines above it list the functions that called this function,
 and the lines below it list the functions this one called.
 This line lists:
     index  A unique number given to each element of the table.
    Index numbers are sorted numerically.
    The index number is printed next to every function name so
    it is easier to look up where the function is in the table.

     % time This is the percentage of the 'total' time that was
    spent
    in this function and its children.  Note that due to
    different viewpoints, functions excluded by options, etc,
    these numbers will NOT add up to 100%.

     self This is the total amount of time spent in this function.

     children This is the total amount of time propagated into
    this
    function by its children.

     called This is the number of times the function was called.
    If the function called itself recursively, the number
    only includes non-recursive calls, and is followed by
    a '+' and the number of recursive calls.

     name The name of the current function.  The index number is
    printed after it.  If the function is a member of a
    cycle, the cycle number is printed between the
    function's name and the index number.


 For the function's parents, the fields have the following
    meanings:

     self This is the amount of time that was propagated directly
    from the function into this parent.

     children This is the amount of time that was propagated from
    the function's children into this parent.

     called This is the number of times this parent called the
    function '/' the total number of times the function
    was called.  Recursive calls to the function are not
```

```
201     included in the number after the '/'.
202
203      name This is the name of the parent.  The parent's index
204     number is printed after it.  If the parent is a
205     member of a cycle, the cycle number is printed between
206     the name and the index number.
207
208  If the parents of the function cannot be determined, the word
209  '<spontaneous>' is printed in the 'name' field, and all the other
210  fields are blank.
211
212  For the function's children, the fields have the following
       meanings:
213
214      self This is the amount of time that was propagated directly
215     from the child into the function.
216
217      children This is the amount of time that was propagated from
       the
218     child's children to the function.
219
220      called This is the number of times the function called
221     this child '/' the total number of times the child
222     was called.  Recursive calls by the child are not
223     listed in the number after the '/'.
224
225      name This is the name of the child.  The child's index
226     number is printed after it.  If the child is a
227     member of a cycle, the cycle number is printed
228     between the name and the index number.
229
230  If there are any cycles (circles) in the call graph, there is an
231  entry for the cycle-as-a-whole.  This entry shows who called the
232  cycle (as parents) and the members of the cycle (as children.)
233  The '+' recursive calls entry shows the number of function calls
       that
234  were internal to the cycle, and the calls entry for each member
       shows,
235  for that member, how many times it was called from other members
       of
236  the cycle.

237
238 Copyright (C) 2012-2022 Free Software Foundation, Inc.
239
240 Copying and distribution of this file, with or without
       modification,
241 are permitted in any medium without royalty provided the copyright
242 notice and this notice are preserved.

243
244 Index by function name
245
246   [26] __static_initialization_and_destruction_0(int, int) [20]
       node::~node() [21] std::chrono::duration<double, std::ratio<1l,
        1000l> >::duration<double, void>(double const&)
247   [25] split(node*, node*&, node*&, char*) [17] std::chrono::
       time_point<std::chrono::_V2::system_clock, std::chrono::
       duration<long, std::ratio<1l, 10000000001> > >::
```

```
time_since_epoch() const [22] std::chrono::duration<long, std::
ratio<1l, 10000000001> >::duration<long, void>(long const&)
248 [3] genKey(char*)          [18] std::chrono::duration<double,
std::ratio<1l, 1000l> >::count() const [23] std::common_type<
std::chrono::duration<long, std::ratio<1l, 10000000001> >, std
::chrono::duration<long, std::ratio<1l, 10000000001> > >::type
std::chrono::operator-<std::chrono::_V2::system_clock, std::
chrono::duration<long, std::ratio<1l, 10000000001> >, std::
chrono::duration<long, std::ratio<1l, 10000000001> > >(std::
chrono::time_point<std::chrono::_V2::system_clock, std::chrono
::duration<long, std::ratio<1l, 10000000001> > > const&, std::
chrono::time_point<std::chrono::_V2::system_clock, std::chrono
::duration<long, std::ratio<1l, 10000000001> > > const&)
249 [4] insert(node*&, node*)  [16] std::chrono::duration<long, std
::ratio<1l, 10000000001> >::count() const [24] std::common_type
<std::chrono::duration<long, std::ratio<1l, 10000000001> >, std
::chrono::duration<long, std::ratio<1l, 10000000001> > >::type
std::chrono::operator-<long, std::ratio<1l, 10000000001>, long,
 std::ratio<1l, 10000000001> >(std::chrono::duration<long, std
::ratio<1l, 10000000001> > const&, std::chrono::duration<long,
std::ratio<1l, 10000000001> > const&)
250 [2] search(node*, char*)    [6] std::enable_if<std::chrono::
__is_duration<std::chrono::duration<double, std::ratio<1l, 1000
l> > >::value, std::chrono::duration<double, std::ratio<1l,
1000l> > >::type std::chrono::duration_cast<std::chrono::
duration<double, std::ratio<1l, 1000l> >, long, std::ratio<1l,
10000000001> >(std::chrono::duration<long, std::ratio<1l,
10000000001> > const&) [9] _init
251 [5] destroy(node*)          [7] std::chrono::duration<double,
std::ratio<1l, 1000l> > std::chrono::__duration_cast_impl<std::
chrono::duration<double, std::ratio<1l, 1000l> >, std::ratio<1l
, 1000000l>, double, true, false>::__cast<long, std::ratio<1l,
10000000001> >(std::chrono::duration<long, std::ratio<1l,
10000000001> > const&) [1] main
252 [19] node::node(char*, unsigned long) [8] std::chrono::duration<
 double, std::ratio<1l, 1000l> >::duration<long, std::ratio<1l,
 10000000001>, void>(std::chrono::duration<long, std::ratio<1l,
 10000000001> > const&)
```

5 Выводы

При выполнении лабораторной работы было изучено профилирование, крайне необходимое для качественной разработки, изучены возможные методы работы с ним. Были использованы утилиты Valgrind для контроля утечек памяти, а также gprof, которая выводит число вызовов функций при работе программы, определяет время работы каждой функции как обособленно, так и в сравнении с общим временем работы программы, что позволяет найти наиболее часто используемую функцию и в первую очередь оптимизировать именно её.