Aiden Pratt
6015 Lab 7

cade lab user name u1458057
ssh u1458057@lab1-11.eng.utah.edu

q1:
this indicates that none of the functions had a significant amount of time to run the program
q2:
Each sample counts as 0.01 seconds.
```
  %   cumulative   self              self    total
 time   seconds   seconds    calls  s/call  s/call  name
 33.09    1.02     1.02 647482750    0.00    0.00 __gnu_cxx::__enable_if<std::__is_char<char>::__value, bool>::__type
std::operator==<char>(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&,
std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
 16.87    1.54     0.52     2    0.26    0.44 sort1(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >*, int)
 15.25    2.01     0.47 647482750    0.00    0.00 bool std::operator!=<char, std::char_traits<char>, std::allocator<char>
>(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char,
std::char_traits<char>, std::allocator<char> > const&)
 14.60    2.47     0.45 38948    0.00    0.00 search1(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >*, int,
std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >)
 11.35    2.82     0.35 379465206    0.00    0.00 bool std::operator< <char, std::char_traits<char>, std::allocator<char>
>(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char,
std::char_traits<char>, std::allocator<char> > const&)
  8.76    3.09     0.27 108224639    0.00    0.00 std::char_traits<char>::compare(char const*, char const*, unsigned long)
  0.32    3.10     0.01     1    0.01    0.01 _GLOBAL__sub_I__Z7search1PNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEiS4_
  0.00    3.10     0.00 38931    0.00    0.00 void std::swap<char, std::char_traits<char>, std::allocator<char>
>(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >&, std::__cxx11::basic_string<char, std::char_traits<char>,
std::allocator<char> >&)
  0.00    3.10     0.00     2    0.00    1.11 find_print_add_records(std::__cxx11::basic_string<char, std::char_traits<char>,
std::allocator<char> >*, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >*, int, int,
std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >)
  0.00    3.10     0.00     2    0.00    0.00 readFile(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >,
std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >*)
  0.00    3.10     0.00     1    0.00    0.00 _GLOBAL__sub_I__Z8readFileNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEEPS4_
  0.00    3.10     0.00     1    0.00    0.00 __static_initialization_and_destruction_0(int, int)
  0.00    3.10     0.00     1    0.00    0.00 __static_initialization_and_destruction_0(int, int)
---------------------------------------------
```

q3:
enable_if is taking the longest is taking 1.02 seconds
this is 32% of the total run time of 3.10 seconds

q4:
it seems that find_print_add_records is the bottleneck

q5:
the search1 function self seconds is .45 and total calls is 38,946
the self seconds per call is 0.00001155 and the total seconds per call is 0.00

q6:
the find-print-add-records method self seconds is 0.00 and total calls is just 2
self seconds per call is 0.00 and total seconds per call is 1.11

q7:
enable_if is taking the most time in seconds at 1.02

Aiden Pratt
6015 Lab 7

q8: and q9:
the original time 3.10 seconds
using the -O1 flag: 1.88 seconds =39.35% faster than original
using the -O2 flag: 2.07 seconds =33.2% faster than original
using the -O3 flag: 1.81 seconds =41.16% faster than original

q10:
this found that -O3 is the fastest optimization time

q11:

Sort function	Search function	Total execution time
Sort1	Search1	2.07
Sort2	Search1	2.10
Sort3	Search1	0.83
Sort1	Search2	0.87
Sort2	Search2	1.76
Sort3	Search2	0.00

q12:
the combination of sort3 and search2 had the fastest propagation time of 0.00 seconds

q13:
the enhancement percentage of this program with the optimized functions is 100%!