```
Name: Jude Onyia
Student ID: V00947095
Email: judeonyia10@gmail.com
Course: ECE596C
Section: T01

Assignment ID: cpp_arithmetic
Assignment Title: Interval Arithmetic and Applications

Submission Source: https://github.com/uvic-seng475-2020-05/cpp_arithmetic-JudeOn
yia.git
Commit ID: 301c29dfd89ae92f8524e4fb633cefe4c7df11b1

Submitted Files
===============

drwxrwxr-x     135 2020-06-15 00:50 ./app
-rw-rw-r--    4120 2020-06-15 00:50 ./app/delaunay_triangulation.cpp
-rw-rw-r--    2427 2020-06-15 00:50 ./app/test_interval.cpp
-rw-rw-r--    3586 2020-06-15 00:50 ./app/test_kernel.cpp
-rw-rw-r--   22880 2020-06-15 00:50 ./app/triangulation_2.hpp
-rw-rw-r--     740 2020-06-15 00:50 ./CMakeLists.txt
-rw-rw-r--     144 2020-06-15 00:50 ./IDENTIFICATION.txt
drwxrwxr-x      24 2020-06-15 00:50 ./include
drwxrwxr-x      56 2020-06-15 00:50 ./include/ra
-rw-rw-r--    6290 2020-06-15 00:50 ./include/ra/interval.hpp
-rw-rw-r--   11224 2020-06-15 00:50 ./include/ra/kernel.hpp
-rw-rw-r--  113754 2020-06-15 00:50 ./README.pdf

Results
=======

Package          Operation Target           Status
nonprog          generate  ---              OK (0.0s)
interval_orig    generate  ---              OK (0.3s)
interval_orig    configure ---              OK (2.3s)
interval_orig    build     test_interval    OK (0.8s)
interval_orig    build     test_kernel      OK (5.2s)
interval_sane    generate  ---              OK (0.4s)
interval_sane    configure ---              OK (2.1s)
interval_sane    build     test_interval    FAIL (2 0.9s 95L)
interval_sane    build     test_kernel      FAIL (2 4.5s 183L)
deltri_orig      generate  ---              OK (0.4s)
deltri_orig      configure ---              OK (2.1s)
deltri_orig      build     delaunay_triangu OK (8.2s)

Normally, an operation is indicated as having a status of either "OK" or
"FAIL".  A status of "?" indicates that the operation could not be performed
for some reason (e.g., due to an earlier error or being a manual step).
The time (in seconds) required for an operation is denoted by an expression
consisting of a number followed by the letter "s" (e.g., "5.0s").
In the case of a test that consists of multiple test cases, the number of
failed test cases and total number of test cases is expressed as a fraction
(e.g., "10/50" means 10 test cases failed out of 50 test cases in total).
The length (in lines) of the log file generated by an operation is denoted by
an expression consisting of a number followed by the letter "L" (e.g., "10L").
To ascertain the reason for the failure of an operation, check the contents
of the log file provided.
```

```
Legend
======

Package: nonprog
    Nonprogramming exercises

Package: interval_orig
    The code as originally submitted by the student.
Build target: test_interval
    Build the test_interval program.
Build target: test_kernel
    Build the test_kernel program.

Package: interval_sane
    Code with modifications to perform API sanity checking.
Build target: test_interval
    Build the (dummy) test_interval program.
Build target: test_kernel
    Build the (dummy) test_kernel program.

Package: deltri_orig
    The code as originally submitted by the student.
Build target: delaunay_triangulation
    Build the delaunay_triangulation program.
```

```
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake
-S/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source
-B/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/derived --check-build-system
CMakeFiles/Makefile.cmake 0
/usr/bin/gmake  -f CMakeFiles/Makefile2 test_interval
gmake[1]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake
-S/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source
-B/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/derived --check-build-system
CMakeFiles/Makefile.cmake 0
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake -E
cmake_progress_start
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived/CMakeFiles 2
/usr/bin/gmake  -f CMakeFiles/Makefile2 CMakeFiles/test_interval.dir/all
gmake[2]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/usr/bin/gmake  -f CMakeFiles/test_interval.dir/build.make
CMakeFiles/test_interval.dir/depend
gmake[3]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
cd
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived &&
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake -E
cmake_depends "Unix Makefiles"
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived/CMakeFiles/test_interval.dir/DependIn
fo.cmake --color=
Scanning dependencies of target test_interval
gmake[3]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/usr/bin/gmake  -f CMakeFiles/test_interval.dir/build.make
CMakeFiles/test_interval.dir/build
gmake[3]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
[ 50%] Building CXX object CMakeFiles/test_interval.dir/app/test_interval.cpp.o
/home/frodo/public/ugls_lab-4.0.70/bin/c++
-I/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source/include  -pedantic-errors
-frounding-math -std=gnu++17 -o
CMakeFiles/test_interval.dir/app/test_interval.cpp.o -c
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_interval.cpp
```

```
In file included from
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_interval.cpp:4:
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/interval.hpp:116:73:
error: C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y [-Wpedantic]
  116 |  typename interval<real_type>::statistics interval<real_type>::stat_ =
{.indeterminate_result_count = 0, .arithmetic_op_count = 0};
      |
^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/interval.hpp:116:106:
error: C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y [-Wpedantic]
  116 |  typename interval<real_type>::statistics interval<real_type>::stat_ =
{.indeterminate_result_count = 0, .arithmetic_op_count = 0};
      |
                                 ^
gmake[3]: *** [CMakeFiles/test_interval.dir/app/test_interval.cpp.o] Error 1
gmake[3]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
gmake[2]: *** [CMakeFiles/test_interval.dir/all] Error 2
gmake[2]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
gmake[1]: *** [CMakeFiles/test_interval.dir/rule] Error 2
gmake[1]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
gmake: *** [test_interval] Error 2
ERROR: build failed to generate executable test_interval
```

```
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake
-S/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source
-B/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/derived --check-build-system
CMakeFiles/Makefile.cmake 0
/usr/bin/gmake  -f CMakeFiles/Makefile2 test_kernel
gmake[1]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake
-S/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source
-B/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/derived --check-build-system
CMakeFiles/Makefile.cmake 0
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake -E
cmake_progress_start
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived/CMakeFiles 2
/usr/bin/gmake  -f CMakeFiles/Makefile2 CMakeFiles/test_kernel.dir/all
gmake[2]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/usr/bin/gmake  -f CMakeFiles/test_kernel.dir/build.make
CMakeFiles/test_kernel.dir/depend
gmake[3]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
cd
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived &&
/home/frodo/public/ugls_lab-4.0.70/packages/cmake-3.17.1/bin/cmake -E
cmake_depends "Unix Makefiles"
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/derived/CMakeFiles/test_kernel.dir/DependInfo
.cmake --color=
Scanning dependencies of target test_kernel
gmake[3]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
/usr/bin/gmake  -f CMakeFiles/test_kernel.dir/build.make
CMakeFiles/test_kernel.dir/build
gmake[3]: Entering directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
[ 50%] Building CXX object CMakeFiles/test_kernel.dir/app/test_kernel.cpp.o
/home/frodo/public/ugls_lab-4.0.70/bin/c++
-I/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-J
udeOnyia/cktmp/package-interval_sane/source/include  -pedantic-errors
-frounding-math -std=gnu++17 -o
CMakeFiles/test_kernel.dir/app/test_kernel.cpp.o -c
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_kernel.cpp
```

```
In file included from
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:6,
                 from
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_kernel.cpp:3:
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/interval.hpp:116:73:
error: C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@
M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y [-Wpedantic]
  116 │  typename interval<real_type>::statistics interval<real_type>::stat_ =
{.indeterminate_result_count = 0, .arithmetic_op_count = 0};
      │
^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/interval.hpp:116:106:
error: C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@
M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y [-Wpedantic]
  116 │  typename interval<real_type>::statistics interval<real_type>::stat_ =
{.indeterminate_result_count = 0, .arithmetic_op_count = 0};
      │
                                  ^
In file included from
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_kernel.cpp:3:
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:25:35: warning:
declaration âM-^@M-^Xstruct ra::math::indeterminate_resultâM-^@M-^Y does not dec
lare anything
   25 │    using idr = typename ra::math::indeterminate_result;
      │                                   ^~~~~~~~~~~~~~~~~~~~
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:27:32: warning:
declaration âM-^@M-^Xclass CGAL::MP_FloatâM-^@M-^Y does not declare anything
   27 │    using exct = typename CGAL::MP_Float;
      │                                ^~~~~~~
In file included from
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/app/test_kernel.cpp:3:
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:59: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
      │                                                            ^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:89: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
```

```
      │
                ^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:119: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
      │
                                              ^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:157: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
      │

    ^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:195: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
      │

                                          ^
/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Jud
eOnyia/cktmp/package-interval_sane/source/include/ra/kernel.hpp:253:237: error:
C++ designated initializers only available with âM-^@M-^X-std=c++2aâM-^@M-^Y or
âM-^@M-^X-std=gnu++2aâM-^@M-^Y
[-Wpedantic]
  253 │  typename Kernel<Real>::Statistics Kernel<Real>::stat_ =
{.orientation_total_count = 0, .orientation_exact_count = 0,
.preferred_direction_total_count = 0, .preferred_direction_exact_count = 0,
.side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count
= 0};
      │


    ^
gmake[3]: *** [CMakeFiles/test_kernel.dir/app/test_kernel.cpp.o] Error 1
gmake[3]: Leaving directory
‘/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived’
gmake[2]: *** [CMakeFiles/test_kernel.dir/all] Error 2
gmake[2]: Leaving directory
‘/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived’
```

```
gmake[1]: *** [CMakeFiles/test_kernel.dir/rule] Error 2
gmake[1]: Leaving directory
`/home/judeonyia/Documents/ECE596C_Assignments/ECE596C_Assgn_3/cpp_arithmetic-Ju
deOnyia/cktmp/package-interval_sane/derived'
gmake: *** [test_kernel] Error 2
ERROR: build failed to generate executable test_kernel
```

```
commit 2e5f4130237556ce16fe8499c8eb2cd218206a6c
Author: JudeOnyia <60678029+JudeOnyia@users.noreply.github.com>
Date:   Thu Jun 11 01:23:01 2020 -0700

    First Commit

commit 0fd81b4d2225985328adcc597aaf191bb0cb2421
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Thu Jun 11 18:12:27 2020 -0700

    1) wrote indeterminate_result class
    2) wrote rounding_mode_saver class
    3) wrote statistics class
    4) wrote default constructor and compound operator overloads (+,-,*)
    5) wrote two argument constructor
    6) wrote upper, lower, is_singleton and sign member functions

commit 4d3078e339c39e99c68a742c23c559fa9c688e26
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Fri Jun 12 02:43:44 2020 -0700

    1) Wrote clear_statistics member function
    2) Wrote get_statistics member function
    3) Wrote binary add, sub, mult non-member functions
    4) Wrote less than operator overload
    5) Wrote stream inserter

commit e75ac306de5ff98af6a01dcae95bf65a26f5400a
Author: JudeOnyia <60678029+JudeOnyia@users.noreply.github.com>
Date:   Fri Jun 12 15:34:29 2020 -0700

    Starting Kernel class

commit 8d2796f420bc33bacd8221ec3cf952fec7cad7d0
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Fri Jun 12 20:58:07 2020 -0700

    1) Wrote the types needed for the Kernel class
    2) Wrote the Orientation and Oriented_side enum classes
    3) Wrote Statistics struct
    4) Wrote a template function to compute the determinant of a
       3 by 3 matrix
    5) Wrote the orientation member function that uses interval
       arithmetic or exact arithmetic if interval yields indetermine
       result

commit 68ffe60f61a40f2c717223f08d1813940de3ed7c
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Sat Jun 13 01:09:36 2020 -0700

    1) Wrote static function for solving determinant of 4 by 4 matrix
    2) Wrote side_of_oriented_circle member function

commit 4b77c1fe22e0c876a07c3a18482872e696002f97
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Sat Jun 13 18:55:44 2020 -0700

    1) Wrote preferred_direction member function
    2) Wrote is_strictly_convex_quad member function

commit 26168d5d9d4049b1fb342bc49a4ece6f58d15306
Author: Jude Onyia <judeonyia10@gmail.com>
```

```
Date:   Sat Jun 13 23:11:03 2020 -0700

    1) Wrote is_locally_delaunay_edge memeber function
    2) Wrote is_locally_pd_delaunay_edge member function
    3) Wrote clear_statistics member function
    4) Wrote get_statistics member function

commit 5db9ce8ba4110ace1ff6126d5ceda2164c813dee
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Sun Jun 14 00:52:55 2020 -0700

    Added some static specifiers and did some const correctness

commit 87a35d587bab40497e037dcb34c70dbc2a67bb24
Author: JudeOnyia <60678029+JudeOnyia@users.noreply.github.com>
Date:   Sun Jun 14 14:57:03 2020 -0700

    Starting delaunay triangulation

commit f479bf137e763c2e18e916d5c3f47ece25eecdb5
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Sun Jun 14 22:20:15 2020 -0700

    1) Wrote function to check if half edge is flippable
    2) Modified is_strictly_convex function in kernel.hpp

commit 3813050045c41cffeade0ff47c11d88fc6816a55
Author: Jude Onyia <judeonyia10@gmail.com>
Date:   Mon Jun 15 00:31:57 2020 -0700

    1) applied the Lawson local optimization procedure

commit 301c29dfd89ae92f8524e4fb633cefe4c7df11b1
Author: JudeOnyia <60678029+JudeOnyia@users.noreply.github.com>
Date:   Mon Jun 15 00:45:17 2020 -0700

    1) Removed traingulation_2_demo
    2) Commented out cmakelists line of demo
```

Name: Jude Onyia

Student ID: V00947095

Course: ECE 596C

Due Date: June 19, 2020

Assignment 3: Non – Programming Exercise

6.1)

Due to exceptions that could be thrown for several reasons (i.e. division by zero, lack of memory, etc.), performing any necessary clean-up of an object must be put in a finalizer of that object. When exception is thrown, the program is violently removed from that code block, it does not execute any code further down the line from that exception. The only code guaranteed to execute is the code within the finalizer of that object. Therefore, any clean-up necessary must be done in the finalizer.

6.2 a)

During the stack unwinding process, objects are destroyed in the following sequence: die3, die, countdown, hello, i, bjarne, herb, dv, u, z.

6.2 b)

During the stack unwinding process, the only object destroyed is s.

6.3 a)

If there is insufficient memory to allocate for second buffer, an exception will be thrown, and the program will be violently ripped from the function without freeing the first buffer. To guarantee that both buffers are freed if an exception occurs, one can use a class for the buffers, where a char pointer is a data member. Overloads on operator= and operator[] can be used to assign the char pointer to a space in memory and access it, respectively. The finalizer of the class can be used to safely deallocate the memory if an exception is thrown. Both buf1 and buf2 will then be objects of this class. In the incident where buf1 is created and memory runs out while attempting to create buf2, as the exception is thrown, it is guaranteed that the finalizer of buf1 will be called to free up that space before leaving the function.

6.3 b)

If the formatting flags are changed and outputting the integer to the ostream causes an exception to be thrown, the program will be violently ripped from the function without running the code that restores the formatting flag. To prevent this, the line that changes the formatting flag and outputs the integer can be surrounded by a try clause. The body of the associated catch clause can restore the old formatting flags if an exception occurs.

6.3 c)

If the queue has a number of elements in it, and there is insufficient amount of memory left to push another element, q.push_back(value) might throws an exception. If it does and the user of the class does not catch it, the program could terminate without freeing the memory allocated for the elements already in the queue. To prevent this, a finalizer for the class can be explicitly defined to free the queue when called. This will ensure that when an exception is thrown, the memory allocated for elements in the queue are freed before the object is destroyed and the program terminated.

6.5)

The function ‘analyze’ cannot throw an exception because the function has the noexcept specifier. This specifier indicates to the compiler that code for exceptions is not needed for this function, therefore, the compile will not generate code for exceptions for this function. This means the function is incapable of throwing an exception. If an attempt is made to throw an exception in this noexcept function, this will result in a fatal error.

The function ‘doWork’ does not have a noexcept specifier, therefore, it may or may not throw an exception. The compiler will generate code for exceptions for this function. Therefore, this function is capable of throwing an exception.

```
# Specify Minimum Required Version
cmake_minimum_required(VERSION 3.1 FATAL_ERROR)

# Specify Project and Language
project(cpp_arithmetic LANGUAGES CXX)

# Set CXX Flags with the -frounding-math option
set(CMAKE_CXX_FLAGS "-frounding-math")

# Find CGAL Library
find_package(CGAL REQUIRED)

# Set Include directories and libraries
include_directories(include ${CGAL_INCLUDE_DIRS})
link_libraries(${CGAL_LIBRARY} ${GMP_LIBRARIES})

# Add Executable Program
add_executable(test_interval app/test_interval.cpp)
add_executable(test_kernel app/test_kernel.cpp)
#add_executable(triangulation_2_demo app/triangulation_2_demo.cpp app/triangulation_2.hpp)
add_executable(delaunay_triangulation app/delaunay_triangulation.cpp app/triangulation_2.hpp)
```

```cpp
#ifndef INTERVAL_HPP
#define INTERVAL_HPP
#include<cfenv>
#include<stdexcept>
#include<algorithm>
#include<iostream>
namespace ra::math {

    struct indeterminate_result : public std::runtime_error{
        using std::runtime_error::runtime_error;
    };

    class rounding_mode_saver {
        public:
            rounding_mode_saver() : current_round_mode(std::fegetround()) {}
            ~rounding_mode_saver() {
                std::fesetround(current_round_mode);
            }
            rounding_mode_saver(rounding_mode_saver&&) = delete;
            rounding_mode_saver(const rounding_mode_saver&) = delete;
            rounding_mode_saver& operator=(rounding_mode_saver&&) = delete;
            rounding_mode_saver& operator=(const rounding_mode_saver&) = delete;
        private:
            int current_round_mode;
    };

    template<class T>
    class interval{
        public:
            using real_type = T;

            struct statistics{
                // The total number of indeterminate results encountered.
                unsigned long indeterminate_result_count;
                // The total number of interval arithmetic operations.
                unsigned long arithmetic_op_count;
            };

            interval(real_type set_value = real_type(0)) : lower_(set_value), upper_(set_value) {}
            interval(real_type lower, real_type upper) : lower_((lower<upper)? lower:upper), upper_((upper>lower)? upper:lower) {}
            real_type lower() const { return lower_; }
            real_type upper() const { return upper_; }

            static void incr_indeterminate_result_count() { ++(stat_.indeterminate_result_count); }
            static void incr_arithmetic_op_count() { ++(stat_.arithmetic_op_count); }

            interval& operator+=(const interval& obj){
                real_type upper_Temp(0);
                real_type lower_Temp(0);
                rounding_mode_saver rms; // Save the rounding mode that should be restored
                std::fesetround(FE_DOWNWARD);
                lower_Temp = lower_ + obj.lower();
                std::fesetround(FE_UPWARD);
                upper_Temp = upper_ + obj.upper();
                lower_ = lower_Temp;
                upper_ = upper_Temp;
                incr_arithmetic_op_count();
                return *this;
            }
            interval& operator-=(const interval& obj){
                real_type upper_Temp(0);
                real_type lower_Temp(0);
                rounding_mode_saver rms; // Save the rounding mode that should be restored
                std::fesetround(FE_DOWNWARD);
                lower_Temp = lower_ - obj.upper();
                std::fesetround(FE_UPWARD);
                upper_Temp = upper_ - obj.lower();
                lower_ = lower_Temp;
                upper_ = upper_Temp;
                incr_arithmetic_op_count();
                return *this;
            }
            interval& operator*=(const interval& obj){
                real_type upper_Temp(0);
                real_type lower_Temp(0);
                rounding_mode_saver rms; // Save the rounding mode that should be restored
                std::fesetround(FE_DOWNWARD);
                lower_Temp = std::min(std::min((lower_*obj.lower()),(lower_*obj.upper())), std::min((upper_*obj.lower()), (upper_*obj.upper())));
                std::fesetround(FE_UPWARD);
                upper_Temp = std::max(std::max((lower_*obj.lower()),(lower_*obj.upper())), std::max((upper_*obj.lower()), (upper_*obj.upper())));
                lower_ = lower_Temp;
                upper_ = upper_Temp;
                incr_arithmetic_op_count();
                return *this;
            }

            bool is_singleton() const { return (lower_==upper_)? true : false; }

            int sign() const {
                if((lower_ < real_type(0)) && (upper_ < real_type(0))){ return -1; }
                else if((lower_ > real_type(0)) && (upper_ > real_type(0))){ return 1; }
                else if((lower_ == real_type(0)) && (upper_ == real_type(0))){ return 0; }
                else{
                    incr_indeterminate_result_count();
                    throw indeterminate_result("Indeterminate sign of interval");
                }
            }

            static void clear_statistics() {
                stat_.indeterminate_result_count = 0;
                stat_.arithmetic_op_count = 0;
            }

            static void get_statistics(statistics& stat) {
                stat = stat_;
            }

        private:
            real_type upper_;
            real_type lower_;
            static statistics stat_;

```

```
    };

    template<class real_type>
    typename interval<real_type>::statistics interval<real_type>::stat_ = {.indeterminate_result_count = 0, .arithmetic_op_count = 0};

    template<class real_type>
    interval<real_type> operator+(const interval<real_type>& obj_A, const interval<real_type>& obj_B){
        real_type upper_Temp(0);
        real_type lower_Temp(0);
        rounding_mode_saver rms; // Save the rounding mode that should be restored
        std::fesetround(FE_DOWNWARD);
        lower_Temp = obj_A.lower() + obj_B.lower();
        std::fesetround(FE_UPWARD);
        upper_Temp = obj_A.upper() + obj_B.upper();
        interval<real_type> result(lower_Temp,upper_Temp);
        result.incr_arithmetic_op_count();
        return result;
    }

    template<class real_type>
    interval<real_type> operator-(const interval<real_type>& obj_A, const interval<real_type>& obj_B){
        real_type upper_Temp(0);
        real_type lower_Temp(0);
        rounding_mode_saver rms; // Save the rounding mode that should be restored
        std::fesetround(FE_DOWNWARD);
        lower_Temp = obj_A.lower() - obj_B.upper();
        std::fesetround(FE_UPWARD);
        upper_Temp = obj_A.upper() - obj_B.lower();
        interval<real_type> result(lower_Temp,upper_Temp);
        result.incr_arithmetic_op_count();
        return result;
    }

    template<class real_type>
    interval<real_type> operator*(const interval<real_type>& obj_A, const interval<real_type>& obj_B){
        real_type upper_Temp(0);
        real_type lower_Temp(0);
        rounding_mode_saver rms; // Save the rounding mode that should be restored
        std::fesetround(FE_DOWNWARD);
        lower_Temp = std::min(std::min((obj_A.lower()*obj_B.lower()),(obj_A.lower()*obj_B.upper())), std::min((obj_A.upper()*obj_B.lower()), (obj_A.upper()*obj_B.upper())));
        std::fesetround(FE_UPWARD);
        upper_Temp = std::max(std::max((obj_A.lower()*obj_B.lower()),(obj_A.lower()*obj_B.upper())), std::max((obj_A.upper()*obj_B.lower()), (obj_A.upper()*obj_B.upper())));
        interval<real_type> result(lower_Temp,upper_Temp);
        result.incr_arithmetic_op_count();
        return result;
    }

    template<class real_type>
    bool operator<(const interval<real_type>& obj_A, const interval<real_type>& obj_B){
        if( (obj_A.upper()) < (obj_B.lower()) ) { return true; }
        else if( (obj_A.lower()) >= (obj_B.upper()) ) { return false; }
        else{
            interval<real_type>::incr_indeterminate_result_count();
            throw indeterminate_result("Indeterminate less than operator");
        }
    }

    template<class real_type>
    std::ostream& operator<<(std::ostream& outStream, const interval<real_type>& obj){
        outStream<<"["<<(obj.lower())<<","<<(obj.upper())<<"]";
        return outStream;
    }


}
#endif
```

```cpp
#include"ra/interval.hpp"
#include<iostream>

int main(){
    using itvf = typename ra::math::interval<float>;
    using stat = typename ra::math::interval<float>::statistics;
    using std::cout;
    using std::endl;

    // Test default constructor
    itvf obj_A;
    cout<<"obj_A: "<<(obj_A.lower())<<" "<<(obj_A.upper())<<endl;
    // Test 2 value constructor
    itvf obj_B(2.89, 2.91);
    itvf obj_C(3.91,3.89);
    cout<<"obj_B: "<<(obj_B.lower())<<" "<<(obj_B.upper())<<endl;
    cout<<"obj_C: "<<(obj_C.lower())<<" "<<(obj_C.upper())<<endl;
    // Test Compound add, sub, mult
    cout<<"obj_C+=obj_B: "<<((obj_C+=obj_B).lower())<<" "<<(obj_C.upper())<<endl;
    cout<<"obj_C-=obj_B: "<<((obj_C-=obj_B).lower())<<" "<<(obj_C.upper())<<endl;
    cout<<"obj_C*=obj_B: "<<((obj_C*=obj_B).lower())<<" "<<(obj_C.upper())<<endl;
    // Test Aliasing of Compound add, sub, mult
    cout<<"obj_B+=obj_B: "<<((obj_B+=obj_B).lower())<<" "<<(obj_B.upper())<<endl;
    cout<<"obj_B-=obj_B: "<<((obj_B-=obj_B).lower())<<" "<<(obj_B.upper())<<endl;
    cout<<"obj_B*=obj_B: "<<((obj_B*=obj_B).lower())<<" "<<(obj_B.upper())<<endl;
    // Test is_singleton
    itvf obj_D(1.7);
    cout<<"obj_D.is_singleton() must be true: "<<(obj_D.is_singleton())<<endl;
    cout<<"obj_C.is_singleton() must be false: "<<(obj_C.is_singleton())<<endl;
    // Test sign()
    itvf obj_E(-7.335,-7.339);
    itvf obj_F(3.442,3.448);
    itvf obj_G(-0.04,0.12);
    cout<<"Neg sign: "<<(obj_E.sign())<<endl;
    cout<<"Pos sign: "<<(obj_F.sign())<<endl;
    cout<<"Zero no sign: "<<(obj_A.sign())<<endl;
    //cout<<"Indeteminate for sign: "<<(obj_G.sign())<<endl;
    // Test get_statistics and clear_statistics
    stat st;
    obj_C.get_statistics(st);
    cout<<"get_statistics of obj_C: indet="<<(st.indeterminate_result_count)<<" arith="<<(st.arithmetic_op_count)<<endl;
    obj_C.clear_statistics();
    obj_C.get_statistics(st);
    cout<<"clear_statistics of obj_C: indet="<<(st.indeterminate_result_count)<<" arith="<<(st.arithmetic_op_count)<<endl;
    // Test Binary add, sub, and mult
    itvf obj_H = obj_E + obj_F;
    itvf obj_I = obj_E - obj_F;
    itvf obj_J = obj_E * obj_F;
    cout<<"[-7.339,-7.335] + [3.442,3.448]: "<<obj_H<<endl;
    cout<<"[-7.339,-7.335] - [3.442,3.448]: "<<obj_I<<endl;
    cout<<"[-7.339,-7.335] * [3.442,3.448]: "<<obj_J<<endl;
    // Test less than
    itvf obj_L(-7.342,-7.337);
    cout<<"Must be true: "<<(obj_E < obj_F)<<endl;
    cout<<"Must be false: "<<(obj_A < obj_E)<<endl;
    //cout<<"Indeterminate: "<<(obj_L < obj_E)<<endl;




    return 0;
}
```

```
#ifndef KERNEL_HPP
#define KERNEL_HPP
#include <CGAL/Cartesian.h>
#include <CGAL/MP_Float.h>
#include <cstddef>
#include"ra/interval.hpp"

namespace ra::geometry {
    // A geometry kernel with robust predicates.
    template<class R>
    class Kernel {
        public:
            // The type used to represent real numbers.
            using Real = R;

            // The type used to represent points in two dimensions.
            using Point =typename CGAL::Cartesian<R>::Point_2;

            // The type used to represent vectors in two dimensions.
            using Vector = typename CGAL::Cartesian<R>::Vector_2;

            // Type used for interval class
            using itv = typename ra::math::interval<R>;
            // Type used for indeterminate_result class
            using idr = typename ra::math::indeterminate_result;
            // Type used for exact arithmetic
            using exct = typename CGAL::MP_Float;

            // The possible outcomes of an orientation test.
            enum class Orientation : int {
                right_turn = -1,
                collinear = 0,
                left_turn = 1,
            };

            // The possible outcomes of an oriented-side-of test.
            enum class Oriented_side : int {
                on_negative_side = -1,
                on_boundary = 0,
                on_positive_side = 1,
            };

            // The set of statistics maintained by the kernel.
            struct Statistics {
                // The total number of orientation tests.
                std::size_t orientation_total_count;
                // The number of orientation tests requiring exact arithmetic
                std::size_t orientation_exact_count;
                // The total number of preferred-direction tests.
                std::size_t preferred_direction_total_count;
                // The number of preferred-direction tests requiring exact arithmetic
                std::size_t preferred_direction_exact_count;
                // The total number of side-of-oriented-circle tests.
                std::size_t side_of_oriented_circle_total_count;
                // The number of side-of-oriented-circle tests requiring exact arithmetic
                std::size_t side_of_oriented_circle_exact_count;
            };

            // Since a kernel object is stateless, construction and destruction are trivial
            Kernel() = default;
            ~Kernel() = default;

            // The kernel type is both movable and copyable.
            // Since a kernel object is stateless, a copy/move operation is trivial
            Kernel(const Kernel&) = default;
            Kernel& operator=(const Kernel&) = default;
            Kernel (Kernel&&) = default;
            Kernel& operator=(Kernel&&) = default;

            // Member function to compute 3 by 3 determinant
            template<class DR>
            static DR determinant_3_by_3(const DR (&m)[3][3]) {
                DR det = ( (m[0][0]*m[1][1]*m[2][2])+(m[0][1]*m[1][2]*m[2][0])+(m[0][2]*m[1][0]*m[2][1]) ) - ( (m[2][0]*m[1][1]*m[0][2])+(m[2][1]*m[1][2]*m[0][0])+(m[2][2]*m[1][0]*m[0][1]) );
                return det;
            }

            // Member function to compute 4 by 4 determinant
            template<class DDR>
            static DDR determinant_4_by_4(const DDR (&m)[4][4]) {
                DDR sub0[3][3] = {m[1][1],m[1][2],m[1][3],m[2][1],m[2][2],m[2][3],m[3][1],m[3][2],m[3][3]};
                DDR sub1[3][3] = {m[0][1],m[0][2],m[0][3],m[2][1],m[2][2],m[2][3],m[3][1],m[3][2],m[3][3]};
                DDR sub2[3][3] = {m[0][1],m[0][2],m[0][3],m[1][1],m[1][2],m[1][3],m[3][1],m[3][2],m[3][3]};
                DDR sub3[3][3] = {m[0][1],m[0][2],m[0][3],m[1][1],m[1][2],m[1][3],m[2][1],m[2][2],m[2][3]};
                DDR det = (m[0][0]*determinant_3_by_3(sub0)) - (m[1][0]*determinant_3_by_3(sub1)) + (m[2][0]*determinant_3_by_3(sub2)) - (m[3][0]*determinant_3_by_3(sub3));
                return det;
            }

            // Determines how the point c is positioned relative to the
            // directed line through the points a and b (in that order).
            // Precondition: The points a and b have distinct values.
            Orientation orientation(const Point& a, const Point& b, const Point& c) const {
                try{
                    ++(stat_.orientation_total_count);
                    itv ax(a.x()); itv ay(a.y()); itv bx(b.x()); itv by(b.y()); itv cx(c.x()); itv cy(c.y());
                    itv matrix[3][3] = {ax,bx,cx,ay,by,cy,itv(1.0),itv(1.0),itv(1.0)};
                    itv det = determinant_3_by_3(matrix);
                    if( (det.sign()) == -1 ) { return (Orientation::right_turn); }
                    else if( (det.sign()) == 1 ) { return (Orientation::left_turn); }
                    else{ return (Orientation::collinear); }
                }
                catch(const idr& e){
                    ++(stat_.orientation_exact_count);
                    exct matrix[3][3] = {(a.x()),(b.x()),(c.x()),(a.y()),(b.y()),(c.y()),exct(1.0),exct(1.0),exct(1.0)};
                    exct det = determinant_3_by_3(matrix);
                    if(det < exct(0)){ return (Orientation::right_turn); }
                    else if(det > exct(0)){ return (Orientation::left_turn); }
                    else{ return (Orientation::collinear); }
                }
            }

            // Determines how the point d is positioned relative to the
            // oriented circle passing through the points a, b, and c
            // (in that order).
            // Precondition: The points a, b, and c are not collinear.
            Oriented_side side_of_oriented_circle(const Point& a, const Point& b, const Point& c, const Point& d)const{
                try{
                    ++(stat_.side_of_oriented_circle_total_count);
                    itv ax(a.x()); itv ay(a.y()); itv bx(b.x()); itv by(b.y()); itv cx(c.x()); itv cy(c.y());
                    itv dx(d.x()); itv dy(d.y());
                    itv ar((ax*ax)+(ay*ay)); itv br((bx*bx)+(by*by)); itv cr((cx*cx)+(cy*cy));
                    itv dr((dx*dx)+(dy*dy));
                    itv matrix[4][4] = {ax,bx,cx,dx,ay,by,cy,dy,ar,br,cr,dr,itv(1.0),itv(1.0),itv(1.0),itv(1.0)};
                    itv det = determinant_4_by_4(matrix);
                    if( (det.sign()) == -1 ) { return (Oriented_side::on_negative_side); }
                    else if( (det.sign()) == 1 ) { return (Oriented_side::on_positive_side); }
                    else{ return (Oriented_side::on_boundary); }
                }
                catch(const idr& e){
                    ++(stat_.side_of_oriented_circle_exact_count);
                    exct ax(a.x()); exct ay(a.y()); exct bx(b.x()); exct by(b.y());
                    exct cx(c.x()); exct cy(c.y());
                    exct dx(d.x()); exct dy(d.y());
                    exct ar((ax*ax)+(ay*ay)); exct br((bx*bx)+(by*by)); exct cr((cx*cx)+(cy*cy));
                    exct dr((dx*dx)+(dy*dy));
                    exct matrix[4][4] = {ax,bx,cx,dx,ay,by,cy,dy,ar,br,cr,dr,exct(1.0),exct(1.0),exct(1.0),exct(1.0)};
                    exct det = determinant_4_by_4(matrix);
                    if(det < exct(0)){ return (Oriented_side::on_negative_side); }
                    else if(det > exct(0)){ return (Oriented_side::on_positive_side); }
                    else{ return (Oriented_side::on_boundary); }
                }
            }

            // Determines if, compared to the orientation of line
            // segment cd, the orientation of the line segment ab is
            // more close, equally close, or less close to the
            // orientation of the vector v.
            // The value returned is 1, 0, or -1 if, compared to the
            // orientation of cd, the orientation of ab is more close,
            // equally close, or less close to the orientation of v,
            // respectively.
            // Precondition: The points a and b have distinct values; the
            // points c and d have distinct values; the vector v is not
            // the zero vector.
            int preferred_direction(const Point& a,const Point& b,const Point& c,const Point& d,const Vector& v)const{
                try{
                    ++(stat_.preferred_direction_total_count);
                    itv ax(a.x()); itv ay(a.y()); itv bx(b.x()); itv by(b.y());
                    itv cx(c.x()); itv cy(c.y()); itv dx(d.x()); itv dy(d.y());
                    itv vx(v.x()); itv vy(v.y());
                    itv first = ( (dx-cx)*(dx-cx) ) + ( (dy-cy)*(dy-cy) );
                    itv second_inc = ( (bx-ax)*(vx) ) + ( (by-ay)*(vy) );
                    itv second = second_inc * second_inc;
                    itv third = ( (bx-ax)*(bx-ax) ) + ( (by-ay)*(by-ay) );
                    itv fourth_inc = ( (dx-cx)*(vx) ) + ( (dy-cy)*(vy) );
                    itv fourth = fourth_inc * fourth_inc;
                    itv result = (first * second) - (third * fourth);
                    return (result.sign());
                }
                catch(const idr& e){
                    ++(stat_.preferred_direction_exact_count);
                    exct ax(a.x()); exct ay(a.y()); exct bx(b.x()); exct by(b.y());
                    exct cx(c.x()); exct cy(c.y()); exct dx(d.x()); exct dy(d.y());
                    exct vx(v.x()); exct vy(v.y());
                    exct first = ( (dx-cx)*(dx-cx) ) + ( (dy-cy)*(dy-cy) );
                    exct second_inc = ( (bx-ax)*(vx) ) + ( (by-ay)*(vy) );
                    exct second = second_inc * second_inc;
                    exct third = ( (bx-ax)*(bx-ax) ) + ( (by-ay)*(by-ay) );
                    exct fourth_inc = ( (dx-cx)*(vx) ) + ( (dy-cy)*(vy) );
                    exct fourth = fourth_inc * fourth_inc;
                    exct result = (first * second) - (third * fourth);
                    if( result < exct(0) ) { return -1; }
                    else if( result > exct(0) ) { return 1; }
                    else { return 0; }
                }
            }

            // Tests if the quadrilateral with vertices a, b, c, and d
            // specified in CCW order is strictly convex.
            // Precondition: The vertices a, b, c, and d have distinct
            // values and are specified in CCW order.
            bool is_strictly_convex_quad(const Point& a,const Point& b,const Point& c,const Point& d) const {
                Orientation t_1 = orientation(a,b,c);
                Orientation t_2 = orientation(b,c,d);
                Orientation t_3 = orientation(c,d,a);
                Orientation t_4 = orientation(d,a,b);
                //Orientation left = Orientation::left_turn;
                //if( (t_1==left) && (t_2==left) && (t_3==left) && (t_4==left) ) { return true; }
                if( (t_2==t_1) && (t_3==t_1) && (t_4==t_1) ){ return true; }
                else{ return false; }
            }

            // Tests if the flippable edge, with endpoints a and c and
            // two incident faces abc and acd, is locally Delaunay.
            // Precondition: The points a, b, c, and d have distinct
            // values; the quadrilateral abcd must be strictly convex.
            bool is_locally_delaunay_edge(const Point& a,const Point& b,const Point& c,const Point& d)const{
                Oriented_side tst = side_of_oriented_circle(a,b,c,d);
                if(tst == Oriented_side::on_positive_side){ return false; }
                else{ return true; }
            }

            // Tests if the flippable edge, with endpoints a and c and
            // two incident faces abc and acd, has the preferred-directions
            // locally-Delaunay property with respect to the first and
            // second directions u and v.
            // Precondition: The points a, b, c, and d have distinct values;
            // the vectors u and v are not zero vectors; the vectors u and
            // v are neither parallel nor orthogonal.
            bool is_locally_pd_delaunay_edge(const Point& a,const Point& b,const Point& c,const Point& d,const Vector& u,const Vector& v)const{
                Oriented_side tst = side_of_oriented_circle(a,b,c,d);
                if(tst == Oriented_side::on_positive_side){ return false; }
                else if(tst == Oriented_side::on_negative_side){ return true; }
                else{
                    if((preferred_direction(a,c,b,d,u)) > 0) { return true; }
                    else if( ((preferred_direction(a,c,b,d,u))==0) && ((preferred_direction(a,c,b,d,v))>0) ){
                        return true;
                    }
                    else{ return false;}
                }
            }

            // Clear (i.e., set to zero) all kernel statistics.
            static void clear_statistics(){
                stat_.orientation_total_count=0;
                stat_.orientation_exact_count=0;
                stat_.preferred_direction_total_count=0;
                stat_.preferred_direction_exact_count=0;
                stat_.side_of_oriented_circle_total_count=0;
                stat_.side_of_oriented_circle_exact_count=0;
            }

            // Get the current values of the kernel statistics.
            static void get_statistics(Statistics& statistics){
                statistics = stat_;
            }

        private:
            static Statistics stat_;

    };

    template<class Real>
    typename Kernel<Real>::Statistics Kernel<Real>::stat_ = {.orientation_total_count = 0, .orientation_exact_count = 0, .preferred_direction_total_count = 0, .preferred_direction_exact_count = 0, .side_of_oriented_circle_total_count = 0, .side_of_oriented_circle_exact_count = 0};



}
#endif
```

```cpp
#include <CGAL/Cartesian.h>
#include"ra/kernel.hpp"
#include<iostream>

int main(){
    using std::cout;
    using std::endl;
    using knlD = typename ra::geometry::Kernel<double>;
    using ort = typename ra::geometry::Kernel<double>::Orientation;
    using ort_side = typename ra::geometry::Kernel<double>::Oriented_side;
    using stat = typename ra::geometry::Kernel<double>::Statistics;
    using point = typename CGAL::Cartesian<double>::Point_2;
    using vector = typename CGAL::Cartesian<double>::Vector_2;

    knlD obj_A;
    point a(0,0); point b(2,2); point c(2,0); point d(1,1); point e(0,2);
    // Test orientation
    cout<<"Test for right(true): "<<((obj_A.orientation(a,b,c)) == (ort::right_turn))<<endl;
    cout<<"Test for collinear(true): "<<((obj_A.orientation(a,b,d)) == (ort::collinear))<<endl;
    cout<<"Test for left(true): "<<((obj_A.orientation(a,b,e)) == (ort::left_turn))<<endl;
    // Test side_of_oriented_circle
    point ac(0,0); point bc(2,0); point cc(0,2); point dc(1,1); point ec(2,2); point fc(3,3);
    cout<<"Test for on_negative_side(true): "<<((obj_A.side_of_oriented_circle(ac,bc,cc,fc)) == (ort_side::on_negative_side))<<endl;
    cout<<"Test for on_boundary(true): "<<((obj_A.side_of_oriented_circle(ac,bc,cc,ec)) == (ort_side::on_boundary))<<endl;
    cout<<"Test for on_positive_side(true): "<<((obj_A.side_of_oriented_circle(ac,bc,cc,dc)) == (ort_side::on_positive_side))<<endl;
    // Test preferred_direction
    point ap(0,0); point bp(2,2); point cp(2,0); point dp(0,2);
    vector up(2,1); vector vp(1,0); vector wp(-1,2);
    cout<<"Test for more close(true): "<<((obj_A.preferred_direction(ap,bp,cp,dp,up)) == 1)<<endl;
    cout<<"Test for equally close(true): "<<((obj_A.preferred_direction(ap,bp,cp,dp,vp)) == 0)<<endl;
    cout<<"Test for less close(true): "<<((obj_A.preferred_direction(ap,bp,cp,dp,wp)) == -1)<<endl;
    // Test is_strictly_convex_quad
    point asc(0,0); point bsc(1,0); point csc(2,2); point dsc(1,2);
    point anc(0,0); point bnc(1,0); point cnc(2,0); point dnc(2,1);
    cout<<"Test for convexity(true): "<<(obj_A.is_strictly_convex_quad(asc,bsc,csc,dsc))<<endl;
    cout<<"Test for convexity(false): "<<(obj_A.is_strictly_convex_quad(anc,bnc,cnc,dnc))<<endl;
    //Test is_locally_delaunay_edge
    point ad(0,0); point bd(1,0); point cd(1,1); point dd(0,1);
    point adn(0,0); point bdn(5,0); point cdn(5,5); point ddn(2,3);
    cout<<"Test edge for locally delaunay(true): "<<(obj_A.is_locally_delaunay_edge(ad,bd,cd,dd))<<endl;
    cout<<"Test edge for locally delaunay(false): "<<(obj_A.is_locally_delaunay_edge(adn,bdn,cdn,ddn))<<endl;
    // Test is_locally_pd_delaunay_edge
    cout<<"Test edge for pref dir loc delaunay(true): "<<(obj_A.is_locally_pd_delaunay_edge(ad,bd,cd,dd,up,vp))<<endl;
    cout<<"Test edge for pref dir loc delaunay(false): "<<(obj_A.is_locally_pd_delaunay_edge(ad,bd,cd,dd,wp,vp))<<endl;
    // Test get_statistics and clear_statistics
    stat st;
    obj_A.get_statistics(st);
    cout<<"Current statistics: ort tot="<<(st.orientation_total_count)<<" ort ex="<<(st.orientation_exact_count)<<" pd tot="<<(st.preferred_direction_total_count)<<" pd ex="<<(st.preferred_direction_exact_count)<<" crc tot="<<(st.side_of_oriented_circle_total_count)<<" crc ex="<<(st.side_of_oriented_circle_exact_count)<<endl;
    obj_A.clear_statistics();
    obj_A.get_statistics(st);
    cout<<"Cleared statistics: ort tot="<<(st.orientation_total_count)<<" ort ex="<<(st.orientation_exact_count)<<" pd tot="<<(st.preferred_direction_total_count)<<" pd ex="<<(st.preferred_direction_exact_count)<<" crc tot="<<(st.side_of_oriented_circle_total_count)<<" crc ex="<<(st.side_of_oriented_circle_exact_count)<<endl;




}
```

```
#include<iostream>
#include"ra/kernel.hpp"
#include <CGAL/Cartesian.h>
#include "triangulation_2.hpp"
#include<vector>

using Kernel = CGAL::Cartesian<double>;
using kernel = ra::geometry::Kernel<double>;
using Triangulation = trilib::Triangulation_2<Kernel>;
using hEit = Triangulation::Halfedge_iterator;
using hE_hand = Triangulation::Halfedge_handle;
using point = CGAL::Cartesian<double>::Point_2;
using vector = CGAL::Cartesian<double>::Vector_2;

/*
using half_edge = Triangulation::Halfedge_handle;
using vertex = Triangulation::Vertex_handle;
*/

bool is_flippable(hE_hand edg_iter, kernel& obj){
    //if((*edg_iter).is_border_edge()){ return false; }
    //else{
        point a = edg_iter->vertex()->point();
        point b = edg_iter->next()->vertex()->point();
        point c = edg_iter->opposite()->vertex()->point();
        point d = edg_iter->opposite()->next()->vertex()->point();
        return (obj.is_strictly_convex_quad(a,b,c,d));
    //}
}

int main(int argc, char** argv){
    using std::cout;
    using std::endl;
    Triangulation tri(std::cin);
    kernel obj;
    hE_hand tmp_hand;


    std::vector<hE_hand> suspect_list;
    for (auto halfedgeIter = tri.halfedges_begin(); halfedgeIter !=tri.halfedges_end(); ++++halfedgeIter) {
        tmp_hand = &*halfedgeIter;
        if(is_flippable(tmp_hand,obj)){
            suspect_list.push_back(tmp_hand);
        }
    }

    while(!(suspect_list.empty())){
        hE_hand  it = suspect_list.back();
        if(is_flippable(it,obj)){
            point c = it->vertex()->point();
            point d = it->next()->vertex()->point();
            point a = it->opposite()->vertex()->point();
            point b = it->opposite()->next()->vertex()->point();
            if(!(obj.is_locally_pd_delaunay_edge(a,b,c,d,vector(1,0),vector(1,1)))){
                it = tri.flip_edge(it);
                suspect_list.push_back(it->next());
                suspect_list.push_back(it->next()->next());
                suspect_list.push_back(it->opposite()->next());
                suspect_list.push_back(it->opposite()->next()->next());
            }
        }
        suspect_list.pop_back();
    }

    cout<<"Resulting Triangulation"<<endl;
    tri.output_off(cout);

    /*
    int count = 0;
    for (auto halfedgeIter = tri.halfedges_begin(); halfedgeIter !=tri.halfedges_end(); ++++halfedgeIter) {
        if(is_flippable(halfedgeIter,obj)){ ++count; }
        cout<<halfedgeIter->vertex()->point()<<endl;
    }
    cout<<"Count was: "<<count<<endl;
    */



    /*
    vertex other_quad_pt_a; vertex other_quad_pt_b;
    int num_of_edge_faces = 0;
    // Iterate over every edge
    for (auto halfedgeIter = tri.halfedges_begin(); halfedgeIter !=tri.halfedges_end(); ++++halfedgeIter) {
        vertex he_a = halfedgeIter->vertex();
        vertex he_b = halfedgeIter->opposite()->vertex();

        // Check if that edge is flippable
        // iterate over faces
        for (auto faceIter = tri.faces_begin(); faceIter != tri.faces_end(); ++faceIter) {
            half_edge h = faceIter->halfedge();
            vertex face_a = h->vertex();
            vertex face_b = h->opposite()->vertex();
            vertex face_c = h->next()->vertex();
            if( ((face_a==he_a)&&(face_b==he_b)) || ((face_a==he_b)&&(face_b==he_a)) ){
                ++num_of_edge_faces;
                if(num_of_edge_faces == 1){ other_quad_pt_a = face_c; }
                if(num_of_edge_faces == 2){ other_quad_pt_b = face_c; }

            }
            else if( ((face_a==he_a)&&(face_c==he_b)) || ((face_a==he_b)&&(face_c==he_a)) ){
                ++num_of_edge_faces;
                if(num_of_edge_faces == 1){ other_quad_pt_a = face_b; }
                if(num_of_edge_faces == 2){ other_quad_pt_b = face_b; }
            }
            else if( ((face_b==he_a)&&(face_c==he_b)) || ((face_b==he_b)&&(face_c==he_a)) ){
                ++num_of_edge_faces;
                if(num_of_edge_faces == 1){ other_quad_pt_a = face_a; }
                if(num_of_edge_faces == 2){ other_quad_pt_b = face_a; }
            }
            else{}
        }
        // If edge is attached to 2 faces, check if quad is convex (if edge is flippable)
        if(num_of_edge_faces == 2){
            if(obj.is_strictly_convex_quad((he_a.point()),(other_quad_pt_a.point()),(he_b.point()),(other_quad_pt_b.point()))){
                // It is flippable
            }
        }
        num_of_edge_faces = 0;

    }
    */
}



/*
OFF
5 4 0
-1 -1 0
1 -1 0
1 1 0
-1 1 0
0 0 1
3 0 1 4
3 1 2 4
3 2 3 4
3 0 4 3
*/


/*
OFF
9 8 0
-1 -1 0
0 -1 0
1 -1 0
1 0 0
1 1 0
0 1 0
-1 1 0
-1 0 0
0 0 0
3 0 1 8
3 0 8 7
3 1 2 3
3 1 3 8
3 3 4 8
3 8 4 5
3 8 5 7
3 7 5 6
 */
```