GDB (WSL or Linux)/LLDB (Mac) Instructions:

Copy your **native** compile-run.sh file into a new file called compile-run-test.sh, add a -g flag, and replace the ./(filename) line with gdb/lldb (filename):

For example: g++ -O3 -DNDEBUG -march=native -Wall -Wno-unused-function -std=c++17 -Isignalgp-lite/third-party/Empirical/include/ -Isignalgp-lite/include/ -g native.cpp -o sgp_lab
./sgp_lab
gdb sgp_lab or lldb sgp_lab

When you have made these changes, run your compile-run-test.sh, then in your terminal, type:
run (GDB) or process launch (LLDB)
Once you hit your segmentation fault, type:
bt
This will produce a backtrace of your code, which will help you with your debugging.

This will likely be helpful enough by itself, but if you need more information, you can put stops on lines you're concerned about before you run:
break Org.h:42 (GDB) or breakpoint set --file Org.h --line 42 (LLDB)
Then you can continue through your breaks, and when you get to the break you're interested in, print your variables out:
print pop[3]

When you're done and want to quit, type:
q

There's a lot more you can do with GDB, and it is your primary weapon, blade shining in the murky c++ moonlight, capable of reckoning with the eldritch horrors that are seg faults. Use it wisely, and use it often.

For a very useful resource comparing GDB and LLDB commands, here's a link:
https://lldb.llvm.org/use/map.html.