**SageMath**

SageMath (or Sage) is an open-source mathematics software system that integrates many mathematical tools into one unified interface. It aims to provide a comprehensive environment for solving mathematical problems, offering features such as:

1. **Algebra**: Sage supports symbolic computation, such as solving equations, factoring, and expanding expressions.
2. **Calculus**: It can handle differentiation, integration, limits, and series expansions.
3. **Linear Algebra**: Functions for matrix operations, eigenvalues, and vector spaces are available.
4. **Graph Theory**: Sage includes tools for working with graphs, including algorithms for traversal, connectivity, and shortest paths.
5. **Number Theory**: It supports prime number operations, modular arithmetic, and factorization.
6. **Computation**: Sage integrates with other systems like Maxima, GAP, and R to enhance its capabilities in specialized areas.

SageMath is powerful open source software for scientific computing, and it is important to get familiar with it in these days of AI-based applications. The first part of this series on SageMath will introduce you to its installation and basic usage.

Numerous choices abound if you are interested in learning an advanced computing tool, supplementing your already rich repertoire of traditional and popular programming languages and tools. Nevertheless, I propose an outstanding recommendation for your consideration: SageMath. SageMath is a computer algebra system (CAS) with utilities that span nearly all fields of mathematics. It is free and open source software licensed under GNU GPL. SageMath is very powerful software and an open source alternative to proprietary scientific computing tools like Maple, Mathematica, MATLAB, etc. The initial development of SageMath was undertaken by William Stein, and the inaugural version was released in February 2005

**Free-as-in-speech + Python means:**

When students run up against a bug, they might be able to fix it. Instead of just telling them to sit quietly and be patient until someone else fixes the problem for them, there's a reasonable chance that a good student can be involved in the solution. In education nowadays we encourage students to be "active learners" and so on; there is talk of "discovery-based learning". It is very much in the spirit of these educational philosophies for a student to discover a bug, realize it is a bug, and go about trying to fix it.

**For higher-level students:**

Upper division students in mathematics or similar field can participate in fixing bugs on more meaningful level, since they might understand the algorithms being used or know enough about programming to fix broken code. SageMath is intended for professional-level real-world use, so for such students, working on improving SageMath is real-world experience, useful to the students after graduation. (Who would you rather hire? The guy who did all his assignments, or the one who says, "I fixed bugs and added features to a large software project used by tens of thousands of people"?)

Also, undergrad research is super hot these days, and SageMath allows students to be up and running fast. Students who are not already highly proficient programmers have to spend tons of time learning, say, C, and then waste a lot of time messing around with pointer arithmetic. SageMath allows those students who are more interested in math than malloc() to spend more time thinking about math and less time figuring out why their code segfaults.