Egor Sheremetov

89213729062
egor.sheremetov@yandex.ru

Education

Applied Mathematics and Informatics

2020 - now **Higher School of Economics Campus in St. Petersburg** (Bachelor)

Technologies

- **C++ (11+)**
- **Java**
- **Python**
- **Assembly (x86)**
- **CMake**
- **Bash**
- **Git**
- **Haskell**
- **C language**

Projects

- Co-work (`C++`, `asio`)

  Wrote team-project in C++ using `boost::asio`. I developed asynchronous C++ server that can handle about 2000 requests per second (of course, all client applications were launched locally). Project itself is created for managing team work on daily basis using Kanban methodology.

- L-language parser (`Python`, `pyparsing`)

  Created our own typed language (based on L abstract language description), after that created grammar for it and parsed it, generated AST (using parser combinators libraries) that can be used in future to interpret it and validate types, for instance.

- Regexp engine (`C++`)

  Created regular expression engine that is able to tell whether string matches given pattern or not. All academic regular expressions were supported. Used parsing with derivatives technique.

- Vector (`C++`)

  Analogue of `std::vector` that offers strong exception guarantee (even better than in standard `C++` library).

- Mytest (`C++`)

  Library for writing unit tests in a convenient and simple way.

- Smart pointers implementation (`C++`)

  Implementation of smart pointers (`unique_ptr` and `shared_ptr`).

Current and completed relevant coursework

- **Graph theory** (chromatic numbers, matching in bipartite and general graphs, planar graphs)
- **Calculus** (differentials, integrals, measure theory, complex variable theory)
- **Algebra** (number theory, group theory, polynomials, complex numbers, linear algebra)
- **Algorithms and data structures** (dynamic programming, algorithms on graphs, complexity theory, greedy algorithms, number theory algorithms, persistent data structures)
- **Formal languages** (regular languages, DFAs, context-free grammars)
- **Functional programming** (Haskell, lambda calculus, type inference)
- **Computer architecture and operating systems** (Assembly, C language, calculations on SSE registers)
- **Probability theory**