NATIONAL RESEARCH UNIVERSITY

Static Program Analysis

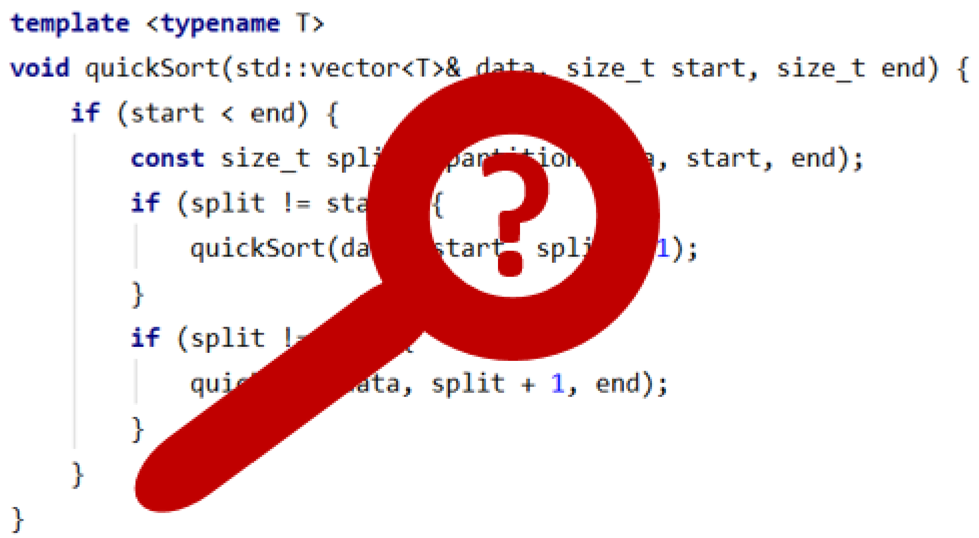
Lecture 1: Introduction. Static Analysis. Software Metrics.

Andrei Tatarnikov

atatarnikov@hse.ru

@andrewt0301

Course Resources



```
template <typename T>
void quickSort(std::vector<T>& data, size_t start, size_t end) {
    if (start < end) {
        const size_t spli[illegible]partition[illegible]a, start, end);
        if (split != sta[illegible]{
            quickSort(da[illegible]start[illegible]spli[illegible]1);
        }
        if (split ![illegible]
            qui[illegible]ata, split + 1, end);
        }
    }
}
```

- **Wiki**

http://wiki.cs.hse.ru/SPA_2022

Web site

https://andrewt0301.github.io/static-analysis-course/

- **Telegram channel**

https://t.me/+gazeL_TRsRYyMWYy

- **Telegram chat**

https://t.me/+d33WiFnng905Mjdi

Course Staff

Andrei Tatarnikov

- HSE page: https://www.hse.ru/org/persons/36585966
- Email: atatarnikov@hse.ru
- Telegram: @andrewt0301
- LinkedIn: https://www.linkedin.com/in/andreitatarnikov/
- GitHub: https://github.com/andrewt0301

Course Outline

Course Contents

https://andrewt0301.github.io/static-analysis-course/

- **15** Lectures and **15** Workshops
- Spoken Exam: **0-10 Points**
- Bonus Points for Presentation at Workshop: **+1 Point Each**

Course Motivation

- Increase your computer literacy
- Improve understanding of languages and compilers
- Understand modern static analysis problems and techniques
- Be able to use static analysis tools in your projects
- Learn how to create static analysis tools

What is Static Program Analysis

Static Program Analysis is a method that allows developers to ensure code quality without running it. Modern software companies use a variety of static program analysis tools and even create their solutions to cover specific requirements.

Any Questions?

```
            .text
__start:    addi t1, zero, 0x18
            addi t2, zero, 0x21
cycle:      beq t1, t2, done
            slt t0, t1, t2
            bne t0, zero, if_less
            nop
            sub t1, t1, t2
            j cycle
            nop
if_less:    sub t2, t2, t1
            j cycle
done:       add t3, t1, zero
```