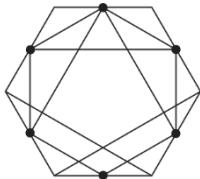
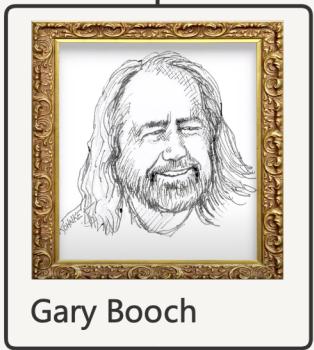


SINGLE SEGREATION PRINCIPLE



Clean Code

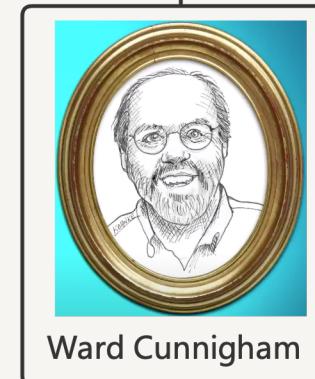


Simple & Direct

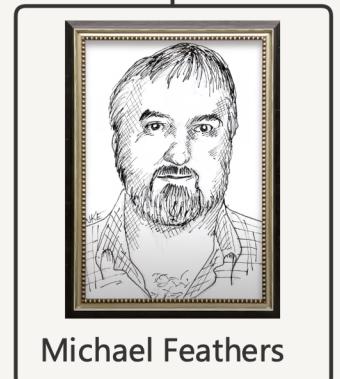
Reads as well-written prose



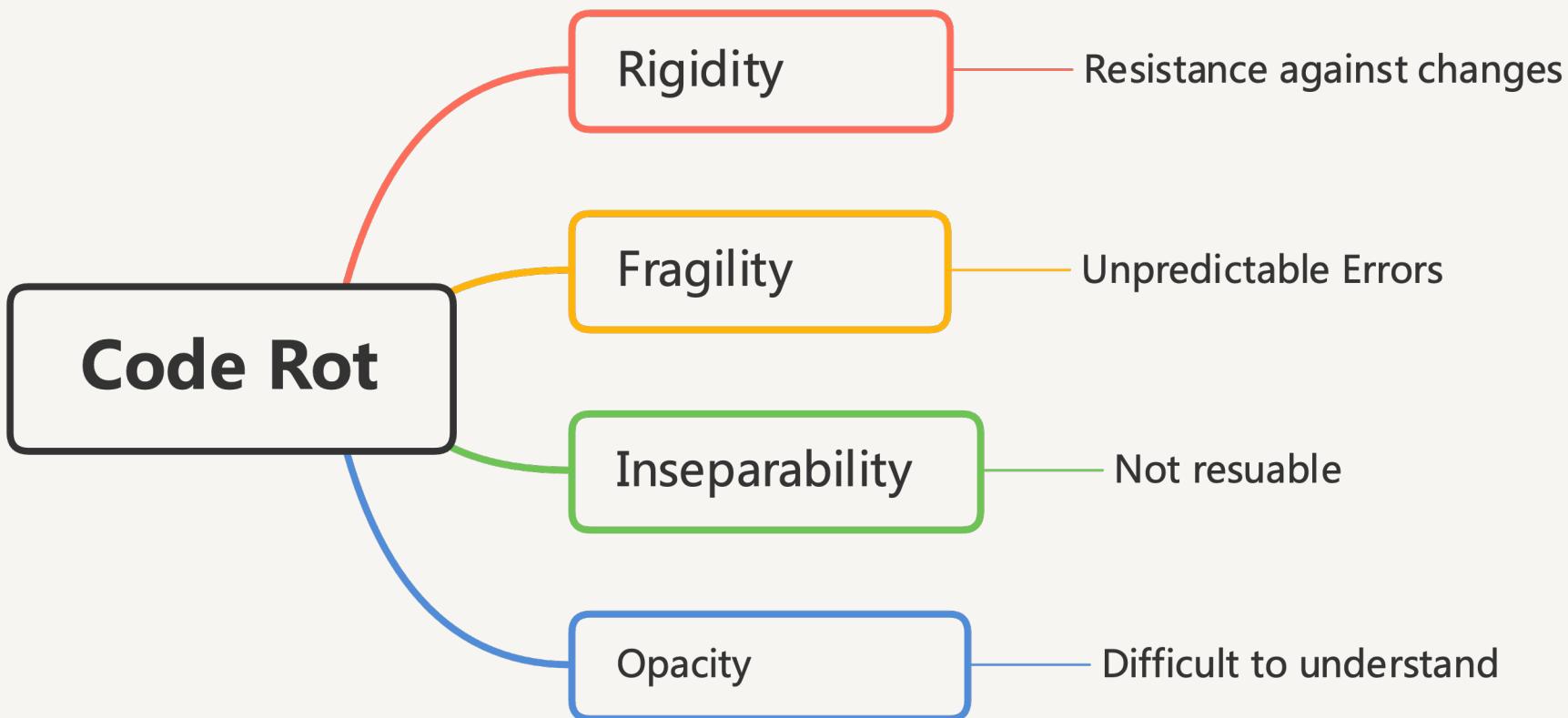
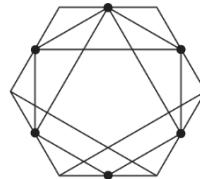
Elegant & Efficient. Should do one thing

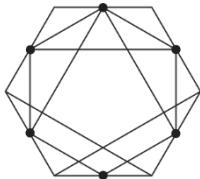


Every Routine is pretty
much what you expected



Written by someone who cares





SOLID

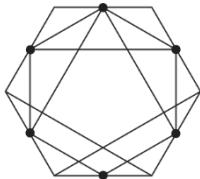
Single Responsibility Principle

Open Close Principle

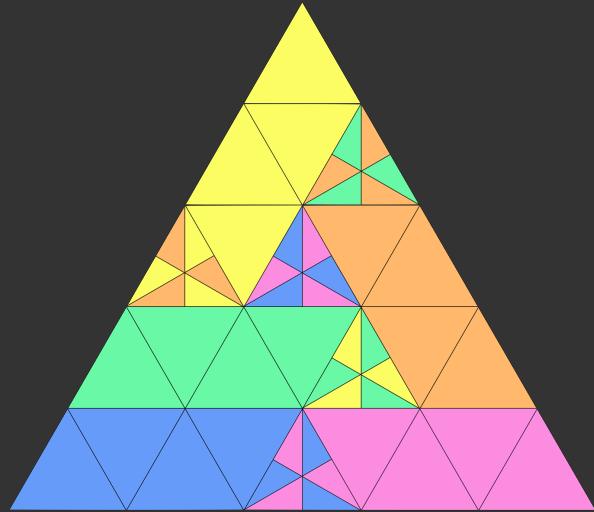
Liskov Substitution Principle

Interface Segregation Principle

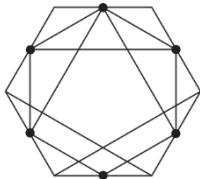
Dependency Inversion Principle



INTERFACE SEGREGATION PRINCIPLE



- Do not depend on things
- You do not need!

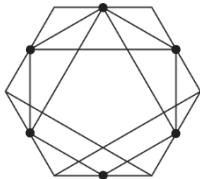


INTERFACE SEGREGATION PRINCIPLE

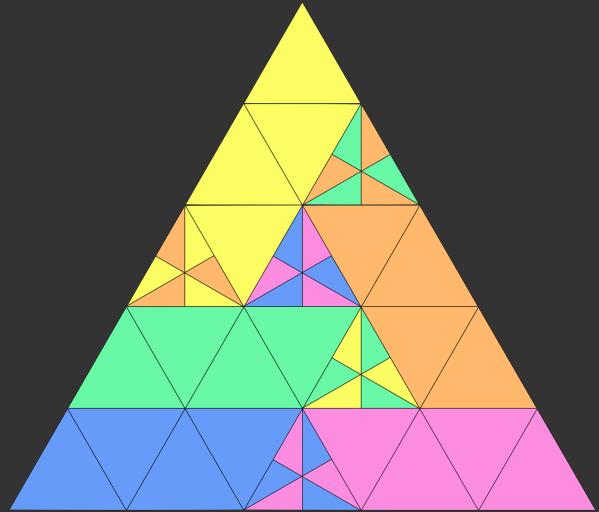
class C



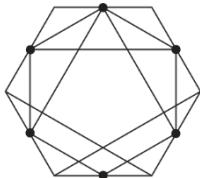
- No client of a class C
- Should depend on methods it doesn't use



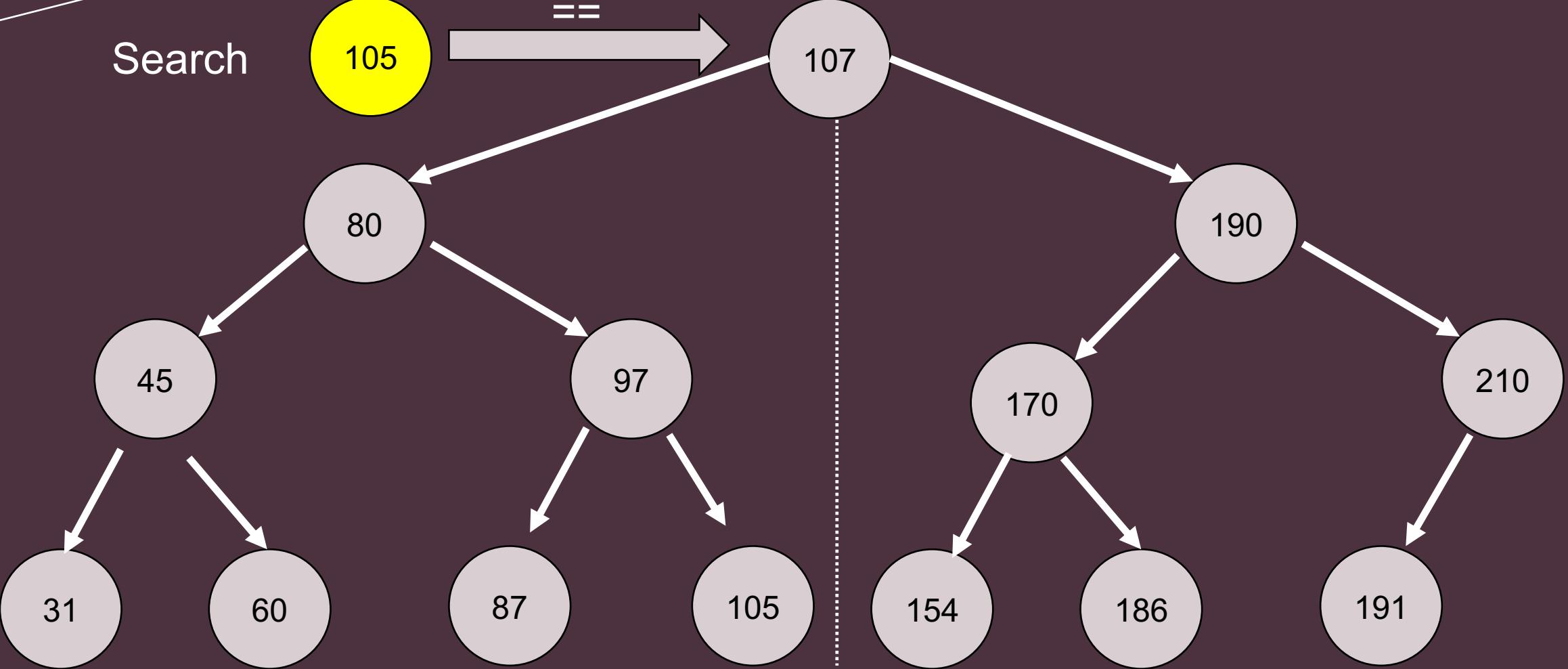
INTERFACE SEGREGATION PRINCIPLE

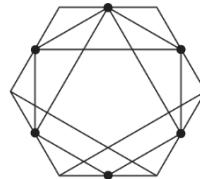


- If that is not the case
- Divide the interface of C into multiple independent interface
- Replace C in each client
- with the interfaces actually used by the client



Search





Search

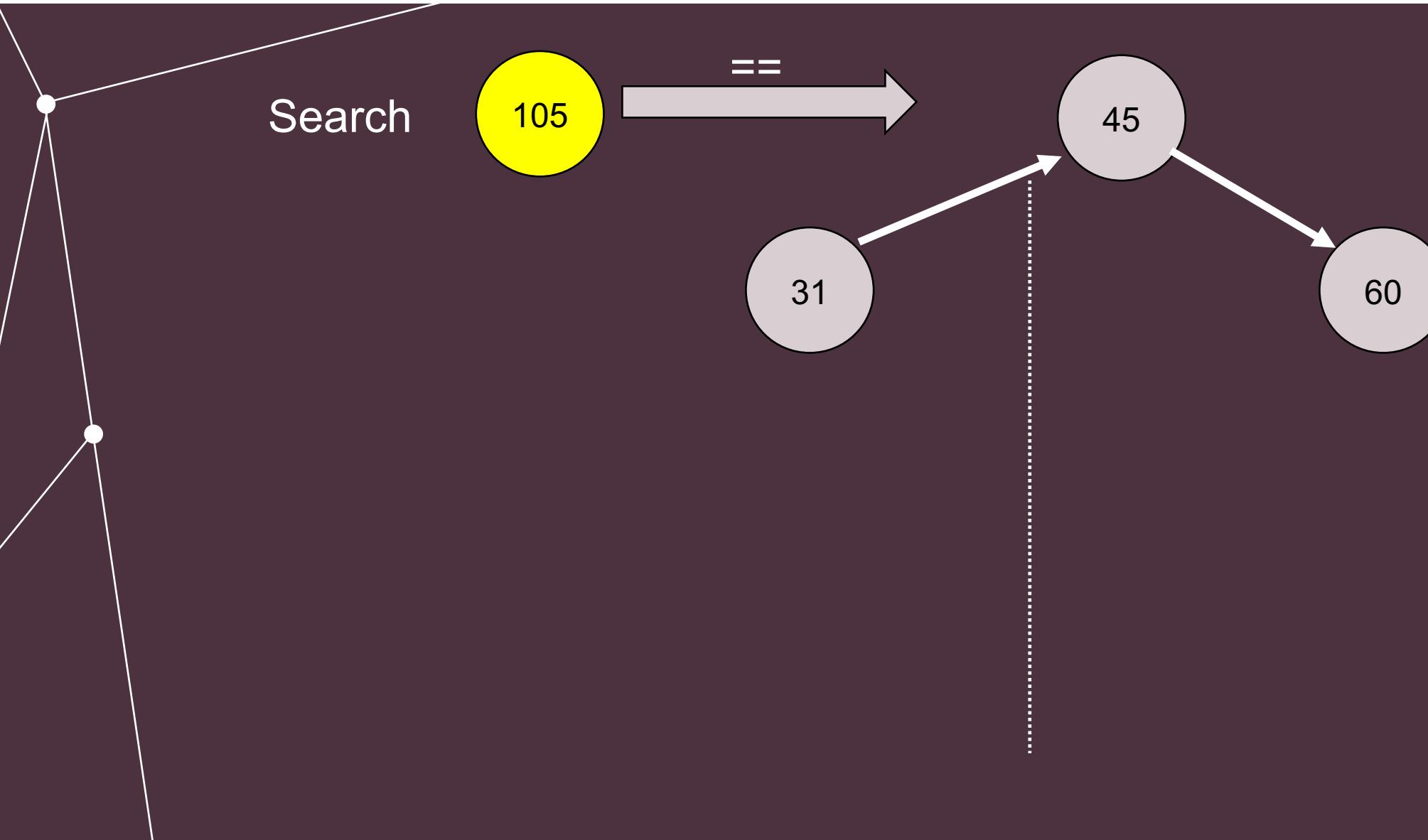
105

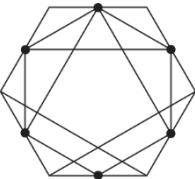


31

45

60

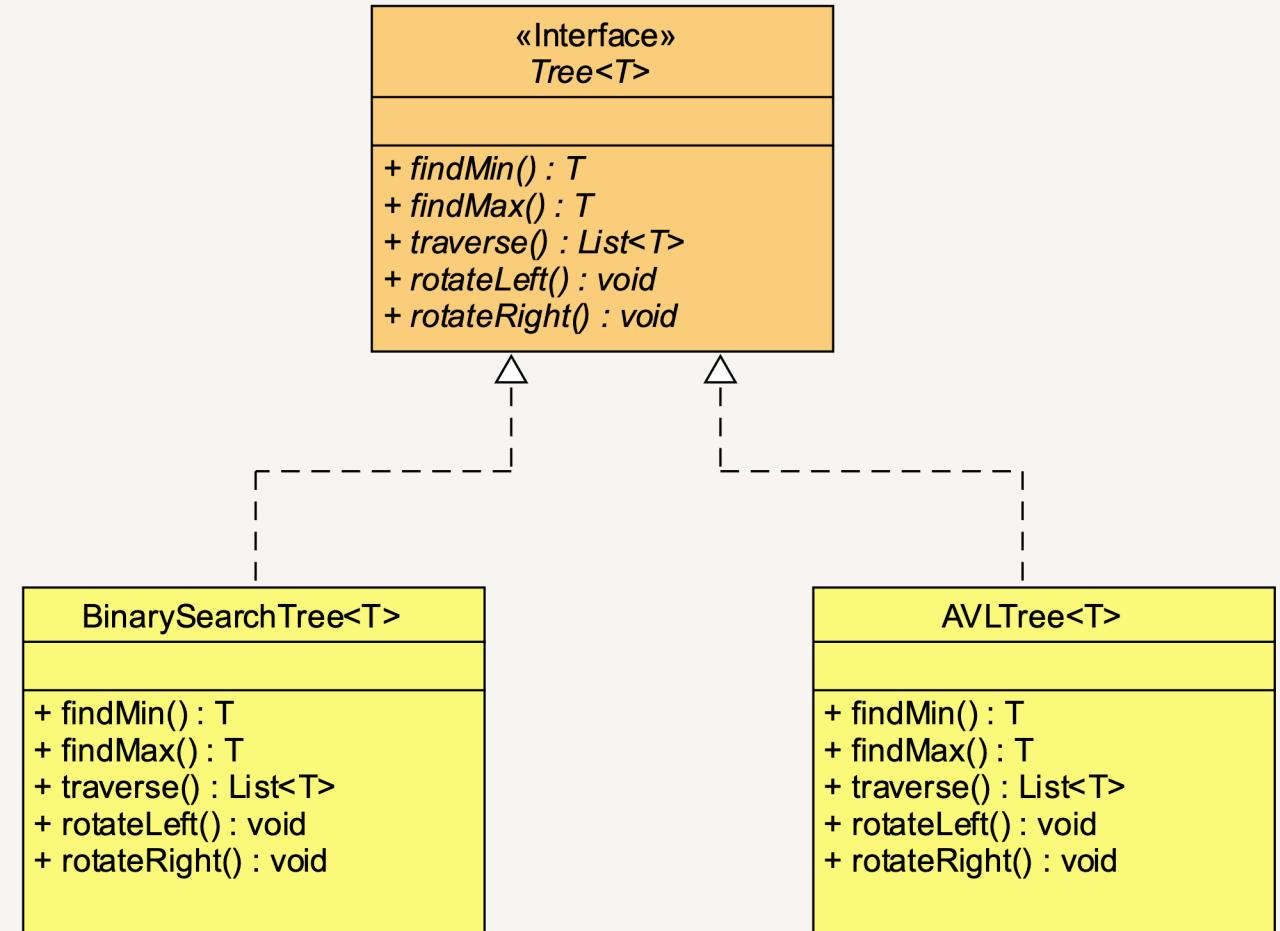
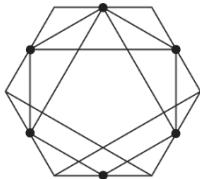




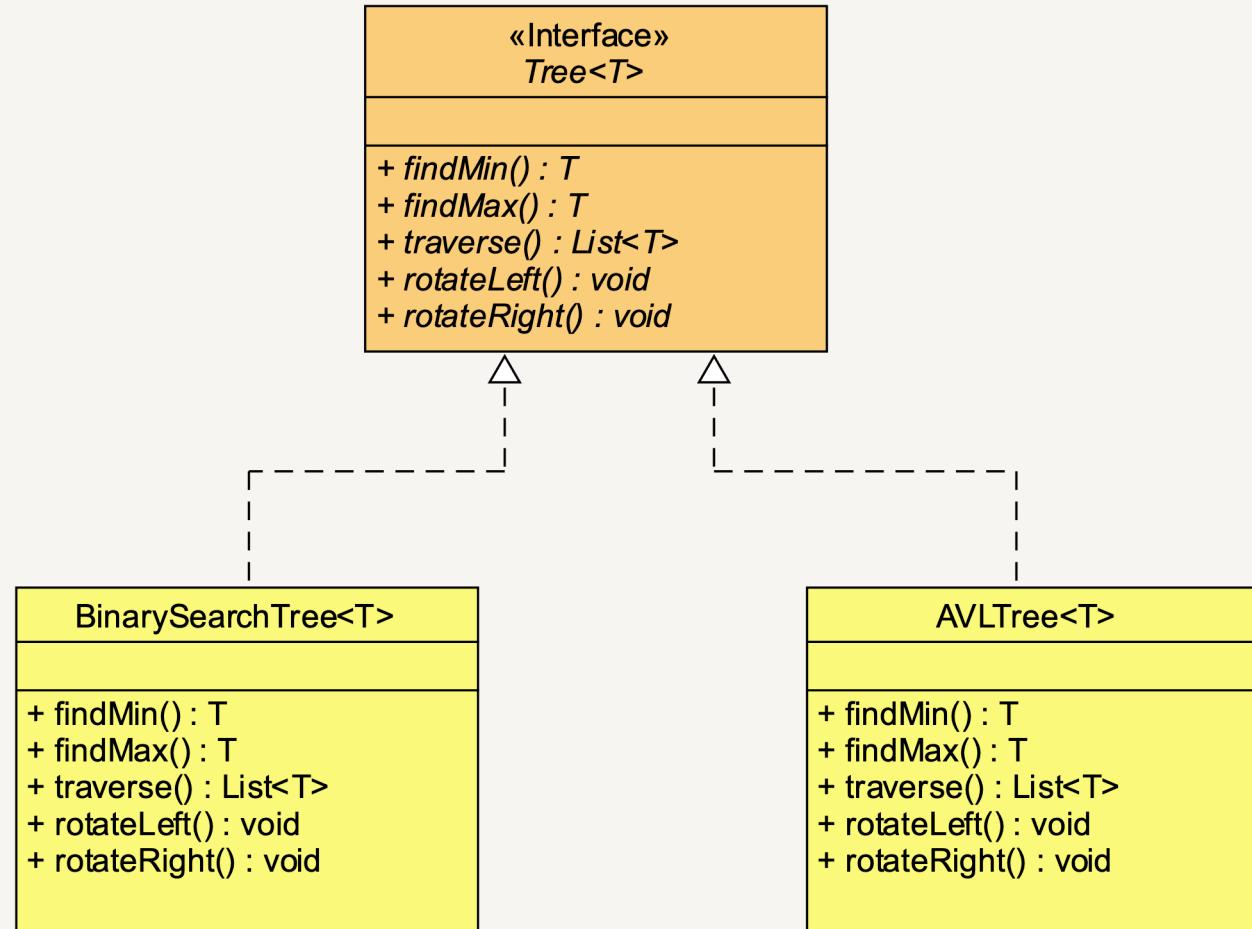
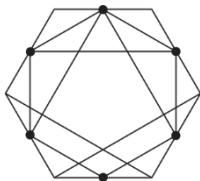
WHY LOG(N)

- In each step we divide in half
- $32 \Rightarrow 16 \Rightarrow 8 \Rightarrow 4 \Rightarrow 2 \Rightarrow 1$
- $32 = 2^5$
- Run-Time $\log(32) = \log(2^5) = 5$

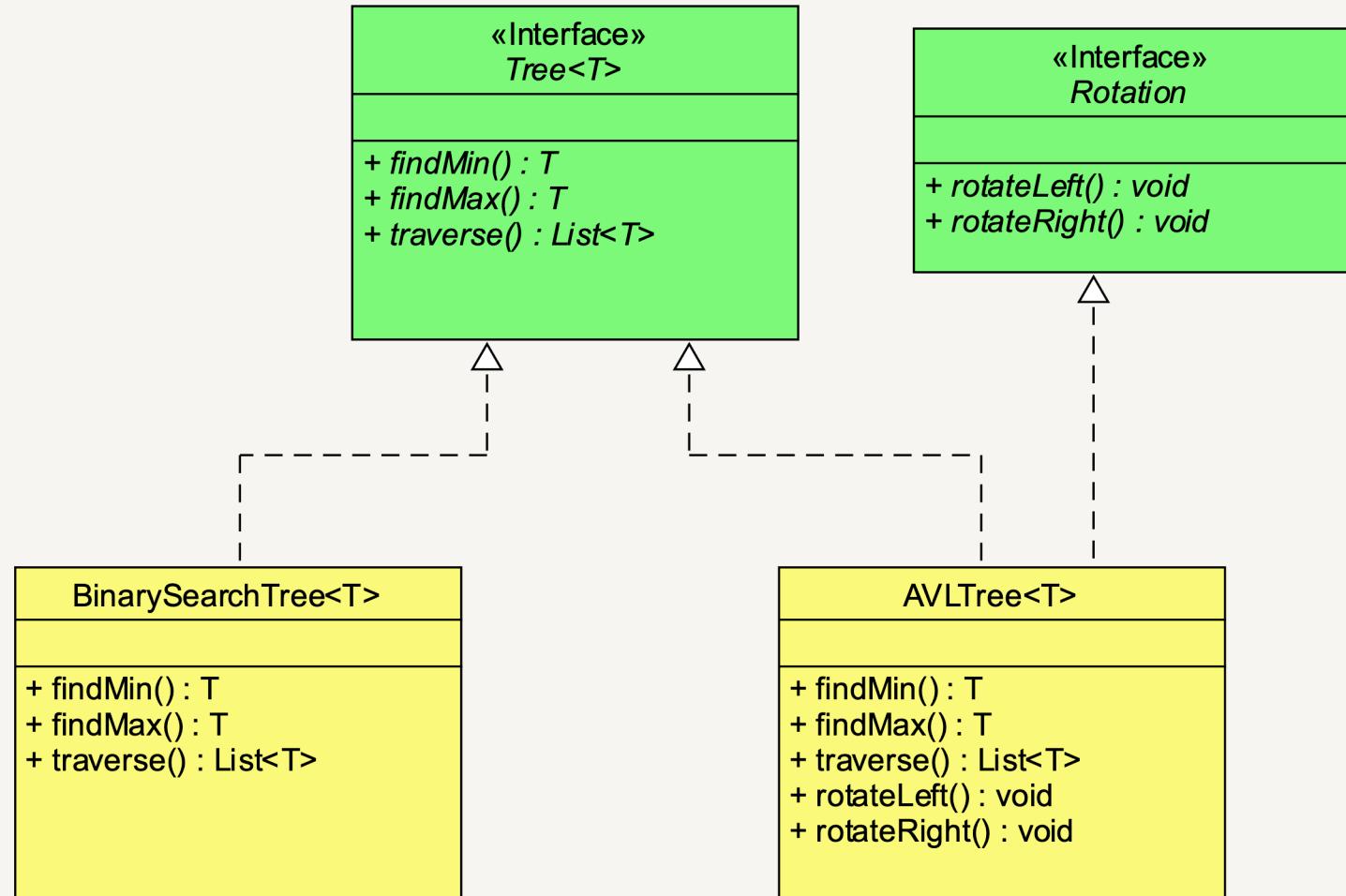
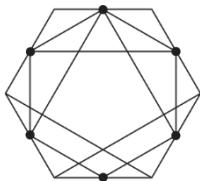
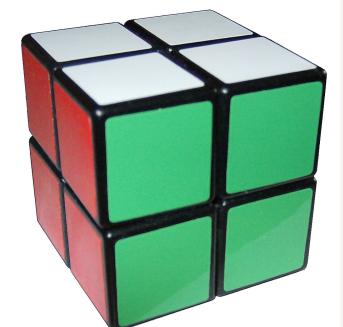


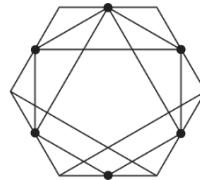


ISP VIOLATION

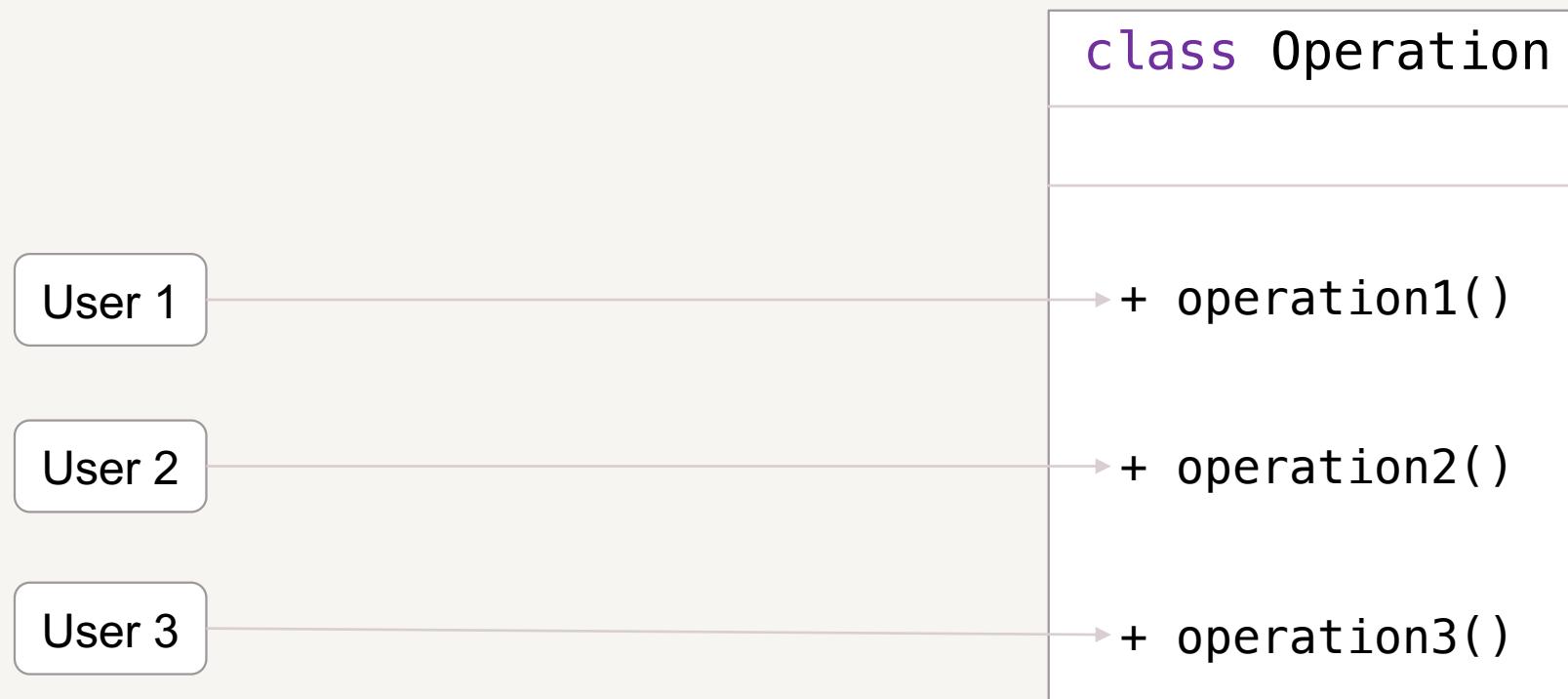


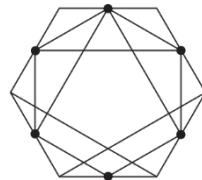
SOLUTION



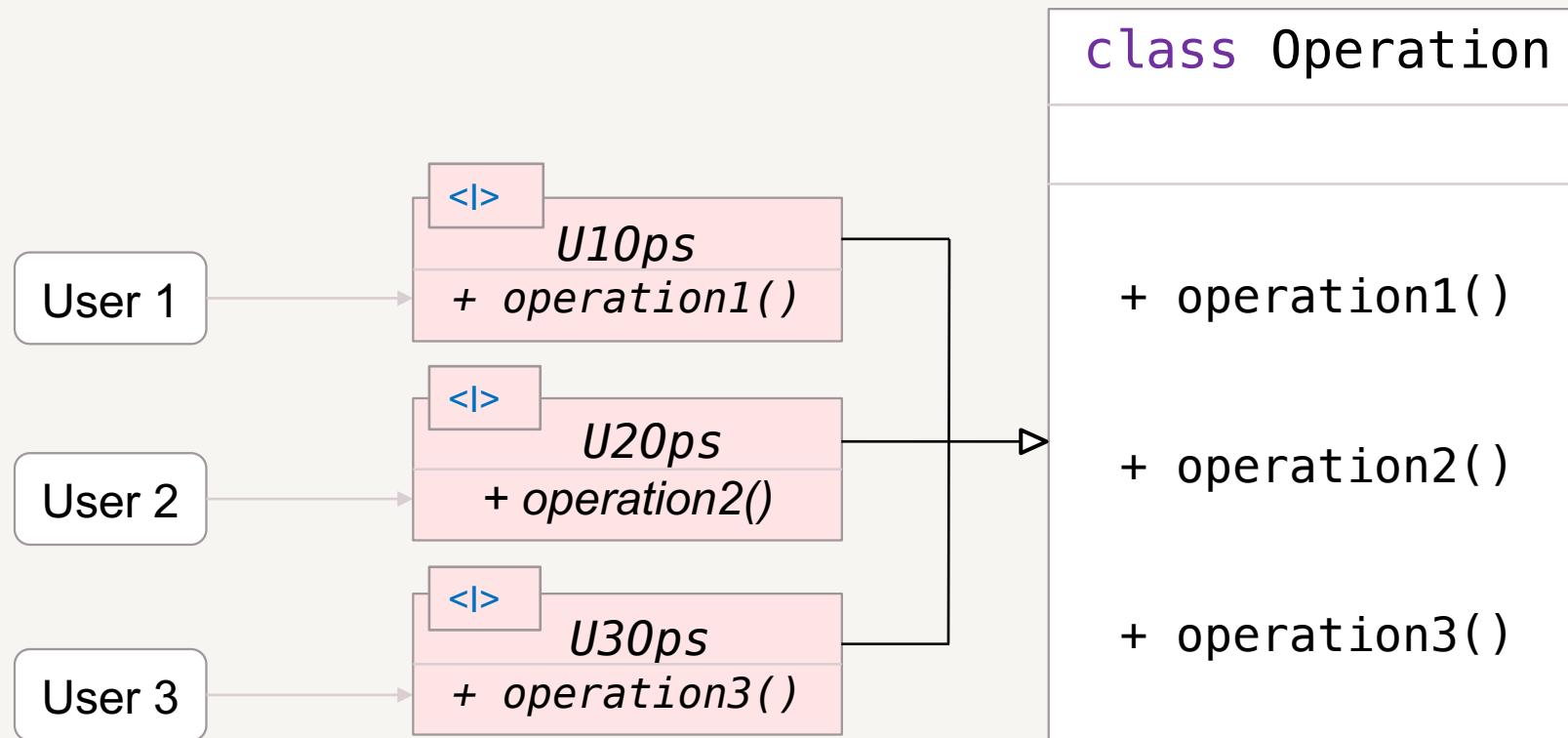


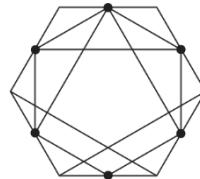
ISP = DO NOT DEPENDS ON THINGS YOU DO NOT NEED!





ISP = DO NOT DEPENDS ON THINGS YOU DO NOT NEED!





ÖSUNGEN UND DISKUSSIONEN





VIELEN DANK FÜR IHRE AUFMERKSAMKEIT.