Ask me anything

2 questions
4 upvotes
What languages have you programmed in?

- Java: 36
- Python: 27
- Ruby: 2
- C++: 12
- R: 9
- Perl: 0
- JavaScript: 9
- Scala: 0
- Smalltalk: 0
- Other: 6
Any languages not mentioned so far?
What is the hardest part about programming?

- Writing clean code
- Right design
- Programming the methods
- First steps
- Changing something is faster
- Coming back to your code
- Finding errors
- Debugging
- Variable naming
- Off-by-1 errors
- Reading written code
- Planning
- Testing
- Splitting up problems
- Global and local vars
- Working with stranger codes
- Figure out what is needed
- Kiss
- Its compact
- Trainings
Procedural

double size() {
    double total = 0;
    for (Shape shape : shapes) {
        switch (shape.kind()) {
            case SQUARE:
                Square square = (Square) shape;
                total += square.width * square.width;
                break;
            case RECTANGLE:
                ...
            case CIRCLE:
                ...
                break;
        }
    }
    return total;
}

Object-oriented

double size() {
    double total = 0;
    for (Shape shape : shapes) {
        total += shape.size();
    }
    return total;
}

public class Square extends Shape {
    ...

    public double size() {
        return width*width;
    }
}
What are the pros and cons of the procedural and OO designs?

**better structure**

**OO con: Fragmentation**

**OO --> easier to change things in an object, without having to think about the rest of the code**

**OO: faster to change, you need to change it once and not in every Methode**

**OO con: can be more time consuming to programm**

**procedural --> everything is in one place, you dont forget to implement something**

**OO: easily add further shapes**

**OO: reusability**

**Procedural: everything together, i.e switch case**

**OO: you can split the programs and test faster certain methods**
What are the pros and cons of the procedural and OO designs?

**OO: access modifiers**
- OO pro: easier to add new methods
- oo: can cause strange errors

**Proc: You can change it individual**
- Oo Its compact, changeing goes faster
- a procedural approach may be closer to one's intuitive way of solving a given problem

**OO: Overloading**
- Procedural: compact codeblock
- proc: easy program flow
What are the pros and cons of the procedural and OO designs?

**OO:** the object can become many many methods, and can become a God Objects

**procedural:** easy to get some quick results for a very specific problem

Because it shows, that the refactoring was not done well

Because it probably could be broken down into more smaller methods

refactoring was not done well
Why are long methods a bad code smell?

- They are incomprehensible.
- Prone to errors
- More than 1 task
- hard for others to understand
- Easily lose the overview.
- more errors during coding
- more difficult to debug
- readability
- it takes more time to understand/debug
Why are long methods a bad code smell?

- Probably take on more tasks than they should
- Likely code duplicates
- They're probably doing multiple steps.
- Slower?
- Have a big area of effect
- Because they could be broken up in easier to understand subtasks.
- Hard to edit without breaking the whole thing
- More opportunities for mistakes
- Harder to reuse
Why are long methods a bad code smell?

- use more space
- refactoring was not done well
- what's up-front design?
Last chance for questions