- Object oriented is outdated for everything to be an object
    - That's why java is bad
    - Garbage collection is a part of classes
    - Has lot of overhead
- Unity is moving away from object oriented for games since it takes a lot to run so performance
- Modularity, abstraction / abstract, encapsulation
    - Modularity
        - Modular building
        - Functions and classes that can be reused elsewhere
        - Small code that can be used over is a little module
        - Notation of module is close to a class
        - Function that fit together in a module
        - If it is not module it is garbage
        - Decomposition
            - How you decompose the software into modules depends on your programming paradigm
                - Decompose into lost of little functions
        - Vector implementation
            - Has three pointers
                - Point at beginning
                - Point at right after end
                - Point capacity
                    - To say how many in array
            - The class countries together the data and the functions that operate on it
            - Either have related functions or classes
                - Those are 2 types of modules
    - Abstraction
        - Opposite of concrete
            - Which means there is an implementation of code there and refers directly to the code
        - Stack is an abstract data type
            - Behavior of the stack is based on implementation
                - What happens if stack is empty
        - Tells you what you can access but not what it does specifically
        - Abstraction is a process
            - How to represent the modules
        - Levels of abstraction
            - If you work close with the facility of machine it is low level abstraction
                - Dealing with memory allocation

                - If it is more of an idea such as making a game with how it works is high level of abstraction
                    - A class of grouping that controls turn order
        - Encapsulation
            - Is about creating boundaries
            - Making nice grouping that have clear what the objects are doing
                - Hiding implementation of things so it can be thought abstractly and not what they are doing concistitly
                - And enforce the idea of not being able to change the implementation
            - You want to make code more abstract to get people to know what it is doing not how it works
            - Generalization
                - It works on more things
                - Overload a function or make it a template to be used for other types extra
                - Specialization is about ballining dowing to be used once
- 3 things we optimize for
    - Correctness
        - Getting the right input output
    - Reduce potential bugs from others
        - Limit access from implementation
        - This is safety
    - Performance
        - It executes prolicialtyl
        - We optimize for this the most
    - Productivity
        - Used to get programmer to work faster with still being done right
    - Usability
        - How easy is it to use