# Notes on DoD

## General Notes:

* Watch polymorphic procedures from Jonathan blow
* OOP encourages AoS (Array of Struct) pattern which is said to be slow

## Research:

* Streaming prefetch bonus and general CPU bonuses
* How to analyze compiler

## Mike Acton talk:

* Where there is one there many (solve the most common problem first)
* The more context you have the better the solution, don’t throw away data or constraints that you need
* OOP confuses Maintenance with Understanding properties of data at the benefit of very little bonus to maintenance the cost is it makes it hard to solve performance problems

## Post Hand in:

### AoS vs SoA

* Maybe try to apply parallelization to AoS vs SoA example to see if there is a bigger boost to one memory access pattern than another
* Experiments
  + Mess around with the iterations
  + Make the same experiment with objects and figure out if there is a measurable difference between thing being stored on the stack vs the heap
  + Introduce more/less data/variables
  + Introduce different kinds of data
  + Introduce operations together with the data (which is typically against the data-oriented philosophy)

### OOP vs DoD example (Stocks)

* Make everything into structs
* Only use flat linear arrays and see if separating huge chunks into separate arrays yield better results
* Look at code metrics

### Disposition

* Introduce the subject
  + Why is it interesting, and what led me to it?
  + How I wanted to approach it
* I really wanted to be able to look at the compiler, why that wasn’t feasible
* Since I couldn’t apply the basic principles or at least the once I was able to find to make a DoD version of Pers Stock example
* Show the project
  + Structs/No classes
  + Linear arrays
  + Data and operation/function/method separation

## References

### Mike Acton Talk

<https://www.youtube.com/watch?v=rX0ItVEVjHc>

### Stoyan Nikolov Talk

<https://www.youtube.com/watch?v=yy8jQgmhbAU>

### Jonathan Blow Talks

<https://www.youtube.com/watch?v=TH9VCN6UkyQ&list=PLmV5I2fxaiCKfxMBrNsU1kgKJXD3PkyxO>

### What Is CPU Cache? Article

<https://www.makeuseof.com/tag/what-is-cpu-cache/>