Understanding Programs
Understanding Swarm

- What objects did the work in Swarm?
- What did those objects need to do that work?
- How did those objects get access to what they needed?
- How do you ensure one object is known, or shared, among many objects?
Variety is Spice of Life

• There are always several ways to do things
  • class
  • function
  • extension
  • variables
  • loops
  • conditional statements

• Each has pros/cons (i.e., trade-offs)
Wrap behavior

- Why separate behavior?
- How to implement?
  - conditionals
  - modular arithmetic
  - use predefined behavior
- Note: what is a container?
Move Randomly Behavior

• How to implement?
  • follow random position
  • turn randomly
  • make random velocity
  • make random changeX, changeY

• Note: how can you have multiple constructors?
Following Behavior

• How to implement?
  • follow leader position exactly
  • follow leader with constrained velocity
• How to get leader’s position?
  • if leader extends shape?
  • if leader does not extend anything?
• How to add randomness?
Changing Color Behavior

• Why not a separate behavior?
• How to implement?
  • based on position
  • based on distance
  • based on size
  • based on cycle
• Note: how to create non-named colors?
Creating Leaders and Followers

• Why make a separate class?
• How to implement?
  • extends shape, and contains shape
  • does not extend shape, but contains shape
  • extends specific shape
  • extends group of shapes, i.e., Composite
• Note: what is this?
Creating Many Objects

• How to implement?
  • write it out directly
  • loop
  • behavior that stops
  • action over Range, **DoForEach**

• How to implement randomness?

• How to implement patterns?
  • using loop counters

• Note: towards using collections?