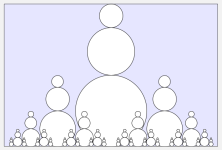
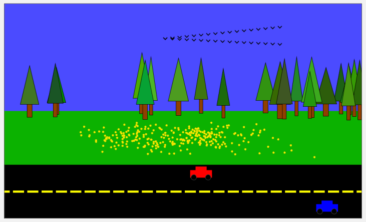
Repetition

Repetition

- self-similarity (accomplished through recursion) provides a form of repetition
  - characterized by similar shapes on smaller scales
  - there is often an increase in the number of copies at each level



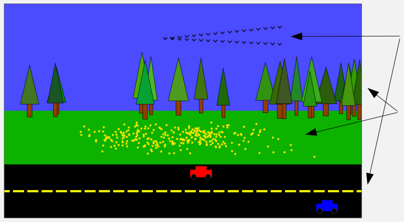
- we might also just want multiple occurrences of elements
  - can create a drawing function and call it multiple times, but even that can be onerous if there are more than a few copies



Loops

When do you need a loop?

- you have more than a few copies of something, and
- what changes between copies changes in a predictable way (or is computed in the same way)

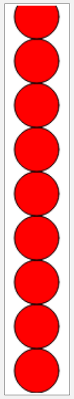


predictable change in position

positions, sizes, colors computed in the same way (random)

Loop Questions

- What is repeated? → loop body
- What changes from one repetition to the next? → loop variables(s)
- How do things start? → initialization of loop variables
- How do things change? → update of loop variables
- When do you keep going? → loop condition
  - may be easier to think “when do you stop?” and then figure out the opposite
  - two patterns
    - repeat as long as / until – condition involves one or more loop variables
    - repeat *n* times (counting loop)



Loop Syntax and Semantics

- for loop

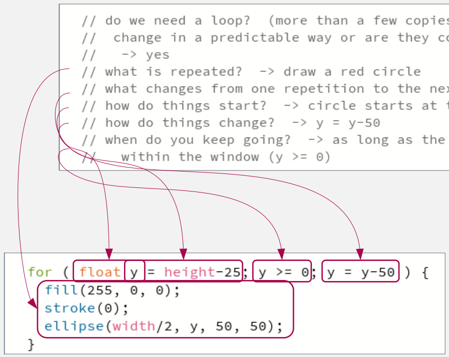
```
for ( declare and initialize loop variables ;
        loop condition ; update loop variables ) {
   loop body
}
```

- while loop

```
declare and initialize loop variables
while ( loop condition ) {
   loop body
   update loop variables
}
```

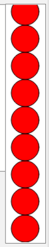
- semantics
  - declare and initialize loop variables
  - evaluate loop condition
    - if true, do the loop body, update loop variables, and evaluate the loop condition again
    - if false, exit the loop

Repeat-As-Long-As Example



```
// do we need a loop?  (more than a few copies?  do the different things
//  change in a predictable way or are they computed in the same way?)
//   -> yes
// what is repeated?  -> draw a red circle
// what changes from one repetition to the next?  -> y coordinate
// how do things start?  -> circle starts at the bottom (y = height-25)
// how do things change?  -> y = y-50
// when do you keep going?  -> as long as the center of the circle is
//   within the window (y >= 0)
```

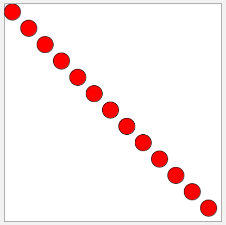
```
for ( float y = height-25; y >= 0; y = y-50 ) {
  fill(255, 0, 0);
  stroke(0);
  ellipse(width/2, y, 50, 50);
}
```



- for loop

```
for ( declare and initialize loop variables ;
        loop condition ; update loop variables ) {
   loop body
}
```

Example



- What is repeated? → loop body
- What changes from one repetition to the next? → loop variables(s)
- How do things start? → initialization of loop variables
- How do things change? → update of loop variables
- When do you keep going? → loop condition
  - may be easier to think "when do you stop?" and then figure out the opposite
  - two patterns
    - repeat as long as / until – condition involves one or more loop variables
    - repeat *n* times (counting loop)

- for loop

```
for ( declare and initialize loop variables ;
        loop condition ; update loop variables ) {
   loop body
}
```