Objects

Programming Lecture 3

Nicholas Dwork

Let's say you're a veterinarian. You'd like a computer program to store all of the cats you've treated.

You want to store their name, age, color, and owner. You also want to store a list of treatments that the cat has received.

For each treatment, you want to store the treatment name, date, and cost.

How can you do this?

Using Arrays

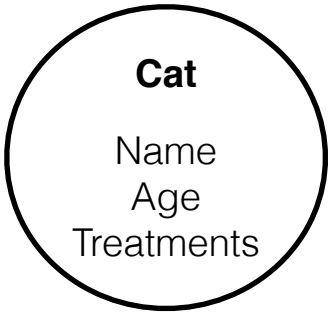
Have many arrays. Use the index to determine which cat we're talking about.

```
catNames = ['felix', 'myCat', 'Garfield'];
catAges = [50, 2, 10];
catTreatments = [ [ immunities, spayed ];
                  [ tumor removal ];
                  [ initial consultation ]; ];
```

This is a very cumbersome approach.
You must always remember to pass around the right information.

Object

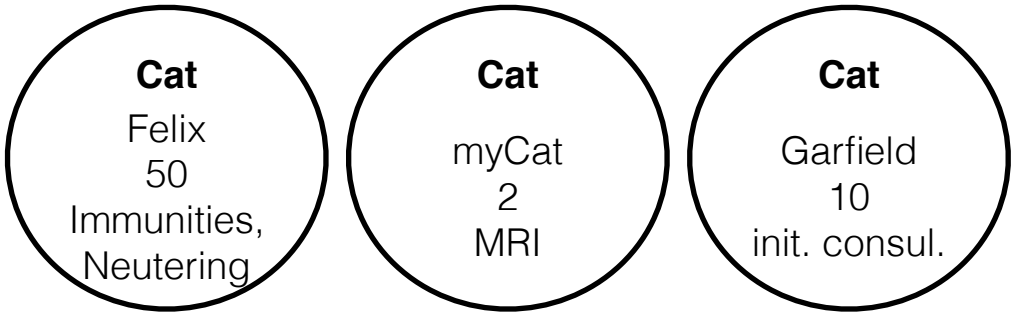
An object allows you to group many variables.



You can then have an array of cats. This makes keeping track of associated values trivial.

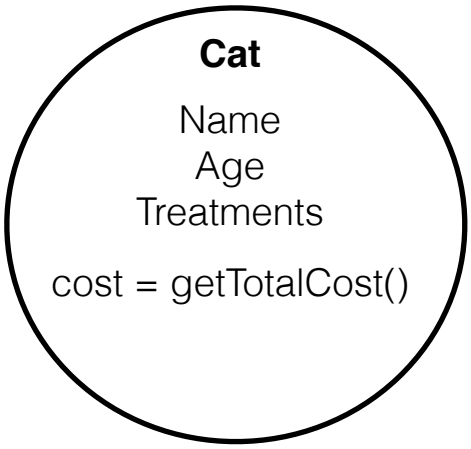
Object

Then you can have several different cats.



Member Functions

An object can have functions inside of it. These functions have access to the variables in the object.



Member Variables

By default, properties are "public" meaning that anyone can change them.
You can change that by setting SetAccess property.

```
classdef cat < handle
   properties ( SetAccess=private )
      name;
      age;
      treatments;
   end
end
```


Constructor

Every object needs a constructor.
This function sets the default values when the object gets instantiated (or created).

```
classdef cat < handle
   …
   methods
      function obj = cat( name, age )
         obj.name = name;
         obj.age = age;
      end
   end
end
```

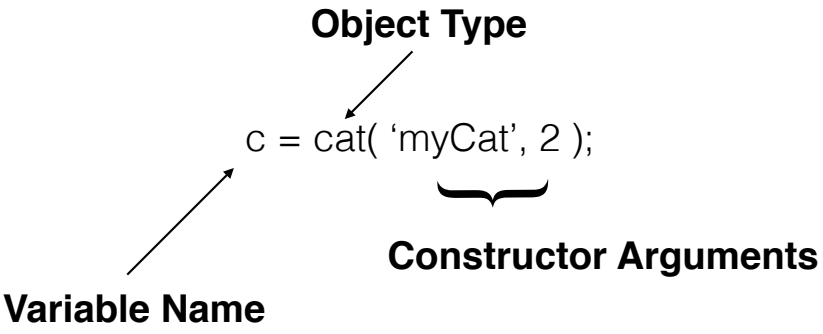
Member Functions

We can add other functions to the cat object.

```
classdef cat < handle
   methods
      function obj = listTreatments( obj )
         disp([‘Treatments for ‘, num2str(obj.name)]);
         for i=1:numel(obj.treatments)
            disp([ ‘   ‘, obj.treatments{i}.name, ‘:  $’, …
               num2str(obj.treatments{i}.cost) ]);
         end
      end
   end
```


Instantiating Objects

We can make variables that store our cats.



Cell Arrays

Objects need to be stored in special arrays called "cell arrays". They work the same as regular arrays, but the syntax is different.

```
cats = cell(3,1);     % create a 3 element cell array

cats{1} = cat('Felix',50);   % make the first element of
                             % cats array a cat.
```