class Hello
  def my_first_method
    puts "Hello, World!"
  end
end

x = Hello.new
x.my_first_method

class A
  def m1
    34
  end
  def m2 (x,y)
    z = 7
    if x > y
      false
    else
      x + y * z
    end
  end
end

class B
  def m1
    4
  end
  def m3 x
    x.abs * 2 + self.m1
  end
end

# example uses for classes A, B, and C (can type into irb)
# here in a multiline comment, which is not well-known
=end

class D
  def m1
    @foo = 0
  end
  def m2 x
    @foo += x
  end
  def foo
    @foo
  end
end

class E
  # uses initialize method, which is better than m1
  # initialize can take arguments too (here providing defaults)
  def initialize(f=0)
    @foo = f
  end
  def m2 x
    @foo += x
  end
end

class F
  # we now add in a class-variable, class-constant, and class-method
  Dans_Age = 41
  def self.reset_bar
    @@bar = 0
  end
  def initialize(f=0)
    @foo = f
  end
  def m2 x
    @foo += x
    @@bar += 1
  end
end

# example uses for classes D, E, and F
=end

Jan 10, 18 14:22 lec19−code−silly.rb Page 3/3

Jan 10, 18 14:22 lec19−code−silly.rb Page 1/3

=begin
a = A.new
thirty_four = a.m1
b = B.new
four = b.m3
forty_seven = B.new.m3 -17
thirty_one = a.m2(3, four)
c = C.new
=end

Jan 10, 18 14:22 lec19−code−silly.rb Page 2/3

c.m1.m2.m3.m1.m1.m3
=end

class D
  def m1
    $foo = 0
  end
  def m2 x
    $foo += x
  end
  def foo
    $foo
  end
end

class E
  # uses initialize method, which is better than m1
  # initialize can take arguments too (here providing defaults)
  def initialize(f=0)
    $foo = f
  end
  def m2 x
    $foo += x
  end
end

class F
  # we now add in a class-variable, class-constant, and class-method
  Dans_Age = 41
  def self.reset_bar
    @@bar = 0
  end
  def initialize(f=0)
    @foo = f
  end
  def m2 x
    @foo += x
    @@bar += 1
  end
end

# example uses for classes D, E, and F
=end

Jan 10, 18 14:22 lec19−code−silly.rb Page 3/3

Jan 10, 18 14:22 lec19−code−silly.rb Page 1/3

=begin
x = D.new
y = D.new # different object than x
z = x # alias to x
x.m2 3 # error because try to add with nil object
x.m1 # creates $foo in object x refers to
=end
z.foo # remember, x and z are aliases
z.m2 3
x.foo
y.m1
y.m2 4
y.foo
x.foo
w = E.new 3
v = E.new
w.foo + v.foo
d = F.new 17
d.m2 5
e = F.new
e.m2 6
d.bar
forty = F::Dans_Age + d.bar
=end

# some examples of adding or changing methods in existing classes
class MyRational # this example extends a class in lec19_example.rb
def double
  self + self
end
end
class Fixnum
def double
  self + self
end
end
def m
  42
end
class Object
def m
  43
end
end