Task 1

my\_num = 25    # Add your code here!

my\_boolean = true    # And here!

my\_string = "Ruby"    # Also here.

puts my\_num

puts my\_boolean

puts my\_string

Task 2

my\_num = 100

# Write code above this line!

puts my\_num

Task 3

x = 2

y = 8

z = 5

ans = y\*\*x-z\*4+x\*y\*z

puts ans

Task 4

puts "Let us commence forth!"

print "Naaaaah"

Task 5

puts "My name is (What?) My name is (Who?) My name is (Chika-chika) Slim Shady".length

Task 6

puts "My name is (What?) My name is (Who?) My name is (Chika-chika) Slim Shady".reverse

Task 7

puts "My name is (What?) My name is (Who?) My name is (Chika-chika) Slim Shady".upcase

puts "My name is (What?) My name is (Who?) My name is (Chika-chika) Slim Shady".downcase

Task 8

# Sample text

Task 9

=begin

Top text

Bottom text

=end

Task 10

name = "Vitaliy"

Task 11

my\_name = "Vitaliy"

my\_age = 19

Task 12

sum = 13 + 379

product = 923 \* 15

quotient = 13209 / 17

puts sum

puts product

puts quotient

Task 13

name = "Vitaliy"

puts name.downcase.reverse.upcase

Task 14

#Finally

=begin

This

was

funky

=end

Chapter 2

Task 1

print "What's your first name?"

Task 2

puts "What's your first name?"

first\_name = gets.chomp

Task 3

print "What's your first name?"

first\_name = gets.chomp

print "What's your last name?"

last\_name = gets.chomp

print "What city are you from?"

city = gets.chomp

print "What state/province are you from?"

state = gets.chomp

Task 4

print "What's your first name?"

first\_name = gets.chomp

print "What's your last name?"

last\_name = gets.chomp

print "What city are you from?"

city = gets.chomp

print "What state/province are you from?"

state = gets.chomp

print "Hello, #{first\_name} #{last\_name} from #{city}, #{state}"

Task 5

print "What's your first name?"

first\_name = gets.chomp

first\_name.capitalize!

print "What's your last name?"

last\_name = gets.chomp

last\_name.capitalize!

print "What city are you from?"

city = gets.chomp

city.capitalize!

print "What state/province are you from?"

state = gets.chomp

state.upcase!

print "Hello, #{first\_name} #{last\_name} from #{city}, #{state}"

CHAPTER 3

Task 1

if 2 > 1

  print "waste of time"

end

Task 2

if 1 > 2

  print "waste of time"

else

  print "still a waste of time"

end

Task 3

if 1 > 2

  print "waste of time"

elsif 1 < 3

  print "still a waste of time"

end

Task 4

hungry = false

unless hungry

  puts "I'm writing Ruby programs!"

else

  puts "Time to eat!"

end

Task 5

is\_true = 2 != 3

is\_false = 2 == 3

Task 6

test\_1 = 17 > 16

test\_2 = 21 < 30

test\_3 = 9 <= 9

test\_4 = -11 < 4

Task 7

# test\_1 = 77 != 77

test\_1 = false

# test\_2 = -4 <= -4

test\_2 = true

# test\_3 = -44 < -33

test\_3 = true

# test\_4 = 100 == 1000

test\_4 = false

Task 8

# boolean\_1 = 77 < 78 && 77 < 77

boolean\_1 = false

# boolean\_2 = true && 100 >= 100

boolean\_2 = true

# boolean\_3 = 2\*\*3 == 8 && 3\*\*2 == 9

boolean\_3 = true

Task 9

# boolean\_1 = 2\*\*3 != 3\*\*2 || true

boolean\_1 = true

# boolean\_2 = false || -10 > -9

boolean\_2 = false

# boolean\_3 = false || false

boolean\_3 = false

Task 10

# boolean\_1 = !true

boolean\_1 = false

# boolean\_2 = !true && !true

boolean\_2 = false

# boolean\_3 = !(700 / 10 == 70)

boolean\_3 = false

Task 11

# boolean\_1 = (3 < 4 || false) && (false || true)

boolean\_1 = true

# boolean\_2 = !true && (!true || 100 != 5\*\*2)

boolean\_2 = false

# boolean\_3 = true || !(true || false)

boolean\_3 = true

Task 11

a = 5

b = 10

if a + b > a \* b

  print "a + b is greater than a \* b"

elsif a + b < a \* b

  print "a + b is less than a \* b"

else

  print "a + b is equal to a \* b"

end

Task 12

dumb = false

print "We are scared of problems not because of their seriousness but because of our perception of them. Got a problem? Ignore it! No more problem" unless dumb

Task 13

# test\_1 should be false

test\_1 = 2 + 2 == 5

# test\_2 = should be false

test\_2 = 123 > 321

# test\_3 = should be true

test\_3 = 2 == 2

Task 14

# test\_1 should be true

test\_1 = (2 + 2 == 4) && (1 + 1 == 2)

# test\_2 = should be true

test\_2 = (9 + 10 == 21) || (1 + 1 == 2)

# test\_3 = should be false

test\_3 = !(1 + 1 == 2)

CHAPTER 4

Task 1 – 8

print "Thith ith dumb "

user\_input = gets.chomp

user\_input.downcase!

if user\_input.include? "s"

  user\_input.gsub!(/s/, "th")

else

  print "No s in the string"

end

puts "#{user\_input}, tah-dah"

CHAPTER 5

Task 1

i = 0

while i < 5

  puts i

  # Add your code here!

  i = i + 1

end

Task 2

counter = 1

until counter > 10

  puts counter

  # Add code to update 'counter' here!

  counter = counter + 1

end

Task 3

counter = 1

while counter < 11

  puts counter

  counter += 1

end

Task 4

for num in 1..15

  puts num

end

Task 5

for i in 1..20

  puts i

end

Task 6

i = 20

loop do

  i -= 1

  print "#{i}"

  break if i <= 0

end

Task 7

i = 20

loop do

  i -= 1

  next if i % 2 == 1

  print "#{i}"

  break if i <= 0

end

Task 8

my\_array = [1, 2, 3, 4, 5]

Task9

odds = [1,3,5,7,9]

# Add your code below!

odds.each { |x|

  print x \* 2

}

Task 10

100.times {print "I HATE THE ANTICHRIST "}

Task 11

i = 1

while i <= 50 do

  print i

  i += 1

end

Task 12

i = 1

until i > 50 do

  print i

  i += 1

end

Task 13

for i in 1..50

  print i

end

Task 14

i = 0

loop do

  print "Ruby!"

  i += 1

  break if i == 30

end

Task 15

30.times {print "Ruby!"}

CHAPTER 5

Task 1-6

puts "gib text"

text = gets.chomp

puts "gib words for SCP wiki moment"

word = gets.chomp

word.downcase!

words = text.split(" ")

redacted = ""

words.each do |wrd|

  if word.include? wrd.downcase

    redacted += "REDACTED "

  else

    redacted += wrd + ' '

  end

end

print redacted

CHAPTER 6

Task 1

my\_array = [69, 420, 21, 1984]

Task 2

demo\_array = [100, 200, 300, 400, 500]

print  demo\_array[2]

Task 3

string\_array = ["Joe Mama", "Moe Lester", "PDF-file"]

Task 4

my\_2d\_array = [[69,69,69],[69,69,69],[69,69,69]]

Task 5

pets = Hash.new

Task 6

pets = Hash.new

pets["Musya"] = "cat"

Task 7

pets = Hash.new

pets["Musya"] = "cat"

puts pets["Musya"]

Task 8

languages = ["HTML", "CSS", "JavaScript", "Python", "Ruby"]

languages.each {|i| puts i}

Task 9

s = [["ham", "swiss"], ["turkey", "cheddar"], ["roast beef", "gruyere"]]

s.each do |sub\_array|

  sub\_array.each {|sandvic| puts sandvic}

end

Task 10

secret\_identities = {

  "The Batman" => "Bruce Wayne",

  "Superman" => "Clark Kent",

  "Wonder Woman" => "Diana Prince",

  "Freakazoid" => "Dexter Douglas"

}

secret\_identities.each do |hero, real\_id|

  puts "#{hero}: #{real\_id}"

end

Task 11

my\_array = [[1, 2, 3], [69, 420, 21], [1488, 1984]]

Task 12

my\_hash = {

  "Joe" => "Mama",

  "Ligma" => "Balls",

  "Deez" => "Nuts"

}

Task 14

lunch\_order = {

  "Ryan" => "wonton soup",

  "Eric" => "hamburger",

  "Jimmy" => "sandwich",

  "Sasha" => "salad",

  "Cole" => "taco"

}

lunch\_order.each {|name, food| puts food}

CHAPTER 7

Task 1 – 7

puts "Give text:"

text = gets.chomp

words = text.split(" ")

frequencies = Hash.new(0)

words.each do |word|

  frequencies[word] += 1

end

frequencies = frequencies.sort\_by do |word, count|

  count

end

frequencies.reverse!

frequencies.each do |word, count|

  puts word + " " + count.to\_s

end

CHAPTER 8

Task 1

def puts\_1\_to\_10

  (1..10).each { |i| puts i }

end

puts\_1\_to\_10 # Ignore this for now. We'll explain it soon!

Task 2

# Define your method below!

def greeting

  puts "Yo, Kiryu-chan!"

end

# Define your method above this line.

greeting # Ignore this for now. We'll explain

         # it in the next exercise!

Task 3

def array\_of\_10

  puts (1..10).to\_a

end

array\_of\_10

Task 4

def cubertino(n)

  puts n \*\* 3

end

cubertino(8)

Task 5

def add(a, b)

  return a + b

end

Task 6

def greeter(name)

  return "Hello, " + name

end

def by\_three?(number)

  return (number % 3 == 0)

end

Task 7

# The block, {|i| puts i}, is passed the current

# array item each time it is evaluated. This block

# prints the item.

[1, 2, 3, 4, 5].each { |i| puts i }

# This block prints the number 5 for each item.

# (It chooses to ignore the passed item, which is allowed.)

[1, 2, 3, 4, 5].each { |i| puts i \* 5 }

Task 8

my\_array = [3, 4, 8, 7, 1, 6, 5, 9, 2]

# Call the sort! method on my\_array below.

# my\_array should then equal [1, 2, 3, 4, 5, 6, 7, 8, 9].

my\_array.sort!

puts my\_array

Task 9

# library sorting code

books = ["Charlie and the Chocolate Factory", "War and Peace", "Utopia", "A Brief History of Time", "A Wrinkle in Time"]

# How might we sort! the books in alphabetical order? (Hint, hint)

books.sort!

Task 10

book\_1 = "A Wrinkle in Time"

book\_2 = "A Brief History of Time"

book\_1 <=> book\_2

Task 11

books = ["Charlie and the Chocolate Factory", "War and Peace", "Utopia", "A Brief History of Time", "A Wrinkle in Time"]

# To sort our books in ascending order, in-place

books.sort! { |firstBook, secondBook| firstBook <=> secondBook }

# Sort your books in descending order, in-place below

books.sort! { |firstBook, secondBook| secondBook <=> firstBook }

Task 12

def welcome

  puts "Welcome to Ruby!"

end

welcome

Task 13

def welcome(name)

  return"Hello, #{name}"

end

welcome("Joey")

Task 14

my\_array = [1, 2, 3, 4, 5]

my\_array.each do |x|

  puts x \* x

end

CHAPTER 9

Task 1-6

def alphabetize(arr, rev = false)

  arr.sort!

  if rev

    arr.reverse!

  else

    return arr

  end

end

numbers = [1, 2, 5, 4, 3]

puts alphabetize(numbers, true)

CHAPTER 10

Task 1

my\_hash = Hash.new

Task 2

matz = { "First name" => "Yukihiro",

  "Last name" => "Matsumoto",

  "Age" => 47,

  "Nationality" => "Japanese",

  "Nickname" => "Matz"

}

matz.each do |key, value|

  puts matz[key]

end

Task 3

no\_nil\_hash = Hash.new("Balls")

Task 4

# Write your code below!

my\_first\_symbol = :balls

Task 5

symbol\_hash = {

  :one => 1,

  :thing => "I don't know why",    # Fill in these two blanks!

  :it => "It doesn't even matter how har you try",

}

Task 6

strings = ["HTML", "CSS", "JavaScript", "Python", "Ruby"]

# Add your code below!

symbols = []

strings.each do |s|

  symbols.push(s.to\_sym)

end

print symbols

Task 7

strings = ["HTML", "CSS", "JavaScript", "Python", "Ruby"]

# Add your code below!

symbols = []

strings.each do |s|

  symbols.push(s.intern)

end

print symbols

Task 8

movies = {

  :Drive => "Drive is a 2011 American action drama film directed by Nicolas Winding Refn.",

  :My\_Life => "A tragic comedy about a software engineering student who has 3 projects due May"

}

Task 9

movies = {

  drive: "Drive is a 2011 American action drama film directed by Nicolas Winding Refn.",

  my\_Life: "A tragic comedy about a software engineering student who has 3 projects due May"

}

Task 10

movie\_ratings = {

  memento: 3,

  primer: 3.5,

  the\_matrix: 5,

  truman\_show: 4,

  red\_dawn: 1.5,

  skyfall: 4,

  alex\_cross: 2,

  uhf: 1,

  lion\_king: 3.5

}

# Add your code below!

good\_movies = movie\_ratings.select {|k, v| v > 3}

Task 11

movie\_ratings = {

  memento: 3,

  primer: 3.5,

  the\_matrix: 3,

  truman\_show: 4,

  red\_dawn: 1.5,

  skyfall: 4,

  alex\_cross: 2,

  uhf: 1,

  lion\_king: 3.5

}

# Add your code below!

movie\_ratings.each\_key {|k| puts k}

CHAPTER 11

Task 1 - 10

movies = {

  Drive: 7.8

}

puts "Input promt"

choice = gets.chomp

case choice

  when "add"

    puts "Enter movie title"

    title = gets.chomp

    if movies[title.to\_sym] == nil

      puts "Enter movie rating"

      rating = gets.chomp

      movies[title.to\_sym] = rating.to\_i

    else

      puts "Movie is already in the hash"

    end

  when "update"

    puts "Enter movie title"

    title = gets.chomp

    if movies[title.to\_sym] == nil

      puts "Error: #{title} is not in the hash"

    else

      puts "Enter new rating"

      rating = gets.chomp

      movies[title.to\_sym] = rating.to\_i

    end

  when "display"

    movies.each {|k, v| puts "#{k}: #{v}"}

  when "delete"

    puts "Enter movie title"

    title = gets.chomp

    if movies[title.to\_sym] != nil

      movies.delete(title.to\_sym)

    else

      puts "#{title} is not in the hash"

    end

  else

    puts "Error: Unknown command"

end

CHAPTER 12

Task 1

# Type your Ruby code below!

puts "Yo'reu" if true

Task 2

# Type your Ruby code below!

puts "Joe" unless !true

Task 3

# Type your Ruby code below!

puts 1 + 1 == 2 ? "1 + 1 is 2" : "1 + 1 is 3"

Task 4

puts "Hello there!"

greeting = gets.chomp

# Add your case statement below!

case greeting

  when "English" then puts "Hello!"

  when "French" then puts "Bonjour!"

  when "German" then puts "Guten Tag!"

  when "Finnish" then puts "Haloo!"

  else puts "I don't know this language!"

end

Task 5

# Write your code on line 2!

favorite\_language ||= "C#"

puts favorite\_language

Task 6

def multiple\_of\_three(n)

  n % 3 == 0 ? "True" : "False"

end

Task 7

my\_array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

my\_array.each {|x| puts x if x % 2 == 0}

Task 8

'L'.upto('P') {|letter| puts letter}

Task 9

age = 26

# Add your code below!

age.respond\_to?(:next)

Task 10

alphabet = ["a", "b", "c"]

alphabet << "d" # Update me!

caption = "A giraffe surrounded by "

caption << "weezards!" # Me, too!

puts caption

Task 11

favorite\_things = ["Ruby", "espresso", "candy"]

puts "A few of my favorite things:"

favorite\_things.each do |thing|

  puts "I love #{thing}!"

end

Task 12

puts "One is less than two!" if 1 < 2

Task 13

puts 1 < 2 ? "One is less than two!" : "One is not less than two."

Task 14

puts "What's your favorite language?"

language = gets.chomp

case language

  when "Ruby"

    puts "Ruby is great for web apps!"

  when "Python"

    puts "Python is great for science."

  when "JavaScript"

    puts "JavaScript makes websites awesome."

  when "HTML"

    puts "HTML is what websites are made of!"

  when "CSS"

    puts "CSS makes websites pretty."

else

  puts "I don't know that language!"

end

Task 15

favorite\_animal ||= "lynx"

Task 16

def square(n)

  n \*= n

end

Task 17

3.times {puts "I'm a refactoring master!"}

CHAPTER 13

require 'prime'   # This is a module. We'll cover these soon!

def first\_n\_primes(n)

  return "n must be an integer." unless n.is\_a? Integer

  return "n must be greater than 0." unless n > 0

  Prime.first n

end

first\_n\_primes(10)

CHAPTER 14

Task 1

# Write your code below!

5.times {puts "I'm a block!"}

Task 2

fibs = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55]

# Add your code below!

doubled\_fibs = fibs.collect {|x| x \* 2}

puts doubled\_fibs

Task 3

def double(n)

  yield(n)

  puts n

end

double(2) {|x| puts x \*= 2}

Task 4

floats = [1.2, 3.45, 0.91, 7.727, 11.42, 482.911]

# Write your code below this line!

round\_down = Proc.new { |x| x.floor}

# Write your code above this line!

ints = floats.collect(&round\_down)

print ints

Task 5

group\_1 = [4.1, 5.5, 3.2, 3.3, 6.1, 3.9, 4.7]

group\_2 = [7.0, 3.8, 6.2, 6.1, 4.4, 4.9, 3.0]

group\_3 = [5.5, 5.1, 3.9, 4.3, 4.9, 3.2, 3.2]

# Complete this as a new Proc

over\_4\_feet = Proc.new { |height| height >= 4 }

# Change these three so that they use your new over\_4\_feet Proc

can\_ride\_1 = group\_1.select(&over\_4\_feet)

can\_ride\_2 = group\_2.select(&over\_4\_feet)

can\_ride\_3 = group\_3.select(&over\_4\_feet)

puts can\_ride\_1

puts can\_ride\_2

puts can\_ride\_3

Task 6

def greeter

  yield

end

phrase = Proc.new {puts "Hello there!"}

greeter(&phrase)

Task 7

hi = Proc.new {puts "Hello!"}

hi.call

Task 8

numbers\_array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

strings\_array = numbers\_array.map(&:to\_s)

puts strings\_array

Task 9

strings = ["leonardo", "donatello", "raphael", "michaelangelo"]

# Write your code below this line!

symbolize = lambda {|x| x.to\_sym}

# Write your code above this line!

symbols = strings.collect(&symbolize)

print symbols

Task 10

my\_array = ["raindrops", :kettles, "whiskers", :mittens, :packages]

# Add your code below!

symbol\_filter = lambda {|x| x.is\_a? Symbol}

symbols = my\_array.select(&symbol\_filter)

puts symbols

Task 11

odds\_n\_ends = [:weezard, 42, "Trady Blix", 3, true, 19, 12.345]

# Add your code below!

ints = odds\_n\_ends.select {|x| x.is\_a? Integer}

puts ints

Task 12 – 13

ages = [23, 101, 7, 104, 11, 94, 100, 121, 101, 70, 44]

# Add your code below!

under\_100 = Proc.new {|x| x < 100 ? true : false}

youngsters = ages.select(&under\_100)

puts youngsters

Task 14 – 15

crew = {

  captain: "Picard",

  first\_officer: "Riker",

  lt\_cdr: "Data",

  lt: "Worf",

  ensign: "Ro",

  counselor: "Troi",

  chief\_engineer: "LaForge",

  doctor: "Crusher"

}

# Add your code below!

first\_half = lambda {|key, value| value < "M" ? true : false}

a\_to\_m = crew.select(&first\_half)

puts a\_to\_m

CHAPTER 15

Task 1 – 4

class Person

  def initialize(name)

    @name = name

  end

end

matz = Person.new("Yukihiro")

Task 5

class MyClass

  $my\_variable = "Hello!"

end

puts $my\_variable

Task 6

class Person

  def initialize(name, age, profession)

    @name = name

    @age = age

    @profession = profession

  end

end

Task 7

class Person

  # Set your class variable to 0 on line 3

  @@people\_count = 0

  def initialize(name)

    @name = name

    # Increment your class variable on line 8

    @@people\_count += 1

  end

  def self.number\_of\_instances

    # Return your class variable on line 13

    @@people\_count

  end

end

matz = Person.new("Yukihiro")

dhh = Person.new("David")

puts "Number of Person instances: #{Person.number\_of\_instances}"

Task 8

class Application

  def initialize(name)

    @name = name

  end

end

# Add your code below!

class MyApp < Application

end

Task 9

class Creature

  def initialize(name)

    @name = name

  end

  def fight

    return "Punch to the chops!"

  end

end

# Add your code below!

class Dragon < Creature

  def fight

    return "Breathes fire!"

  end

end

Task 10

class Creature

  def initialize(name)

    @name = name

  end

  def fight

    return "Punch to the chops!"

  end

end

# Add your code below!

class Dragon < Creature

  def fight

    puts "Instead of breathing fire..."

    super()

  end

end

Task 11

class Message

  def initialize(from, to)

    @from = from

    @to = to

  end

end

Task 12

class Message

  @@messages\_sent = 0

  def initialize(from, to)

    @from = from

    @to = to

    @@messages\_sent += 1

  end

end

Task 13

class Message

  @@messages\_sent = 0

  def initialize(from, to)

    @from = from

    @to = to

    @@messages\_sent += 1

  end

end

my\_message = Message.new("A", "D")

Task 14

class Message

  @@messages\_sent = 0

  def initialize(from, to)

    @from = from

    @to = to

    @@messages\_sent += 1

  end

end

my\_message = Message.new("A", "D")

class Email < Message

  def initialize(subject)

    @subject = subject

  end

end

Task 15

class Message

  @@messages\_sent = 0

  def initialize(from, to)

    @from = from

    @to = to

    @@messages\_sent += 1

  end

end

my\_message = Message.new("A", "D")

class Email < Message

  def initialize(from, to)

    super(from, to)

  end

end

CHAPTER 16

class Computer

  @@users = {}

  def initialize(username, password)

    @username = username

    @password = password

    @files = {}

    @@users[username] = password

  end

  def create(filename)

    time = Time.now

    @files[filename] = time

    puts "#{@username} created a #{@filename} file at #{time}"

  end

  def Computer.get\_users

    @@users

  end

end

my\_computer = Computer.new("Moe Lester", "Fromunda")

CHAPTER 17

Task 1

class Dog

  def initialize(name, breed)

    @name = name

    @breed = breed

  end

end

Task 2

class Dog

  def initialize(name, breed)

    @name = name

    @breed = breed

  end

  public

  def bark

    puts "Woof!"

  end

end

Task 3

class Dog

  def initialize(name, breed)

    @name = name

    @breed = breed

  end

  public

  def bark

    puts "Woof!"

  end

  private

  def id

    @id\_number = 12345

  end

end

Task 4

class Person

  def initialize(name, job)

    @name = name

    @job = job

  end

  attr\_reader :name

  attr\_writer :job

end

Task 5

module MyLibrary

  FAVE\_BOOK = "1984"

end

Task 6

puts Math::PI

Task 7

require 'date'

puts Date.today

Task 8

class Angle

  include Math

  attr\_accessor :radians

  def initialize(radians)

    @radians = radians

  end

  def cosine

    cos(@radians)

  end

end

acute = Angle.new(1)

acute.cosine

Task 9

# Create your module here!

module MartialArts

  def swordsman

    puts "I'm a swordsman."

  end

end

class Ninja

  include MartialArts

  def initialize(clan)

    @clan = clan

  end

end

class Samurai

  include MartialArts

  def initialize(shogun)

    @shogun = shogun

  end

end

Task 10

class Application

  attr\_accessor :status

  def initialize; end

  # Add your method here!

  public

  def print\_status

    puts "All systems go!"

  end

end

Task 11

class Application

  attr\_accessor :status

  def initialize; end

  # Add your method here!

  public

  def print\_status

    puts "All systems go!"

  end

  private

  def password

    12345

  end

end

Task 12

module Languages

  FAVE = "Ruby"

end

Task 13

module Languages

  FAVE = "Ruby"  # This is what you typed before, right? :D

end

class Master

  include Languages

  def initialize; end

  def victory

    puts FAVE

  end

end

total = Master.new

total.victory

Chapter 18

class Account

  attr\_reader :name

  attr\_reader :balance

  def initialize(name, balance = 100)

    @name = name

    @balance = balance

  end

  public

  def display\_balance(pin\_number)

    if pin\_number == pin

      puts "Balance: $#{@balance}"

    else

      puts pin\_error

    end

  end

  def withdraw(pin\_number, amount)

    if pin\_number == pin

      puts "Withdrew #{amount}. New balance: $#{@balance}."

    else

      puts pin\_error

    end

  end

  private

  def pin

    @pin = 1234

  end

  def pin\_error

    "Access denied: incorrect PIN."

  end

end

checking\_account = Account.new("Joe Biden", 10000000)