

```
# Первый шаг

class BookInStock

end

a_book = BookInStock.new
another_book = BookInStock.new

# Добавляем инициализатор

class BookInStock
  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end
end

book_one = BookInStock.new('isbn1', 10)
p book_one
puts book_one

book_two = BookInStock.new('isbn2', 1.5)
p book_two
puts book_two

book_three = BookInStock.new('isbn3', '1.25')
p book_three
puts book_three

# Добавляем преобразование к строке

class BookInStock
  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end
end
```

```

    def to_s
      "ISBN: #{@isbn}, price: #{@price}"
    end
  end
end

book_one = BookInStock.new('isbn1', 10)
p book_one
puts book_one

book_two = BookInStock.new('isbn2', 1.5)
p book_two
puts book_two

book_three = BookInStock.new('isbn3', '1.25')
p book_three
puts book_three

# Доступ к переменным экземпляра

class BookInStock
  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end

  def isbn
    @isbn
  end

  def price
    @price
  end

  def to_s
    "ISBN: #{@isbn}, price: #{@price}"
  end
end

```

```
book_one = BookInStock.new('isbn1', 53.32)
puts "ISBN = #{book_one.isbn}"
puts "price = #{book_one.price}"
```

```
# Доступ с помощью attr_reader
```

```
class BookInStock
  attr_reader :isbn, :price

  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end

  def to_s
    "ISBN: #{@isbn}, price: #{@price}"
  end
end
```

```
book_one = BookInStock.new('isbn1', 53.32)
puts "ISBN = #{book_one.isbn}"
puts "price = #{book_one.price}"
```

```
# Установка значения атрибута
```

```
class BookInStock
  attr_reader :isbn, :price

  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end

  def price=(new_price)
    @price = Float(new_price)
  end

  def to_s
```

```

    "ISBN: #{@isbn}, price: #{@price}"
  end
end

book_one = BookInStock.new('isbn1', 29.99)
puts "ISBN = #{book_one.isbn}"
puts "price = #{book_one.price}"
book_one.price = book_one.price * 0.75
puts "new price = #{book_one.price}"

# Установка значения с помощью attr_accessor

class BookInStock
  attr_reader :isbn
  attr_accessor :price

  def initialize(isbn, price)
    @isbn = isbn
    @price = Float(price)
  end

  def to_s
    "ISBN: #{@isbn}, price: #{@price}"
  end
end

book_one = BookInStock.new('isbn1', 29.99)
puts "ISBN = #{book_one.isbn}"
puts "price = #{book_one.price}"
book_one.price = book_one.price * 0.75
puts "new price = #{book_one.price}"

# Виртуальный атрибут price_in_copeks

class BookInStock
  attr_reader :isbn
  attr_accessor :price

```

```

def initialize(isbn, price)
  @isbn = isbn
  @price = Float(price)
end

def price_in_copeks
  Integer(@price * 100 + 0.5)
end

def price_in_copeks=(copeks)
  @price = copeks / 100.0
end

def to_s
  "ISBN: #{@isbn}, price: #{@price}"
end
end

book = BookInStock.new('isbn1', 125.99)
puts "ISBN = #{book.isbn}"
puts "price = #{book.price}"
puts "price in copeks = #{book.price_in_copeks}"
book.price_in_copeks = 11999
puts "price = #{book.price}"
puts "price in copeks = #{book.price_in_copeks}"

# Интерфейс CsvReader

class CsvReader
  def initialize
    # ...
  end

  def read_in_csv_data(csv_file_name)
    # ...
  end

  def total_value_in_stock

```

```

    # ...
end

def number_of_each_isbn
  # ...
end

reader = CsvReader.new
reader.read_in_csv_data('file1.csv')
reader.read_in_csv_data('file2.csv')

# Аккумуляция данных

require 'csv'

class CsvReader
  def initialize
    @books_in_stock = []
  end

  def read_in_csv_data(csv_file_name)
    CSV.foreach(csv_file_name, headers:true) do |row|
      @books_in_stock << BookInStock.new(row['ISBN'], row['Price'])
    end
  end
end

# Подсчёт общего количества денег

class CsvReader
  def initialize
    @books_in_stock = []
  end

  def read_in_csv_data(csv_file_name)
    CSV.foreach(csv_file_name, headers:true) do |row|
      @books_in_stock << BookInStock.new(row['ISBN'], row['Price'])
    end
  end
end

```

```

    end
  end

  def total_value_in_stock
    sum = 0.0
    @books_in_stock.each {|book| sum += book.price}
    sum
  end
end

# Инициализация приложения

reader = CsvReader.new
ARGV.each do |csv_file_name|
  STDERR.puts "Processing #{csv_file_name}"
  reader.read_in_csv_data(csv_file_name)
end
puts "Total value in stock #{reader.total_value_in_stock}"

# Пример ограничения доступа

class Account
  attr_accessor :balance
  def initialize(balance)
    @balance = balance
  end
end

class Transaction
  def initialize(account_a, account_b)
    @account_a = account_a
    @account_b = account_b
  end

  private

  def debit(account, amount)
    account.balance -= amount
  end
end

```

```
end

def credit(account, amount)
  account.balance += amount
end

public

#...

def transfer(amount)
  debit(@account_a, amount)
  credit(@account_b, amount)
end

#...
end

savings = Account.new(100)
checking = Account.new(200)
trans = Transaction.new(checking, savings)
trans.transfer(50)
```