Robert James Bowes

I & T - IT1

```
public class Card {
    private Enum<Suit> suit;
    private Enum<Value> value;

    public Card(Enum<Suit> suit, Enum<Value> value) {
        this.suit = suit;
        this.value = value;
    }

    Enum<Suit> getSuit() { return suit; }

    Enum<Value> getValue() { return value; }

    int getValueRanking() { return value.ordinal() + 1; }

    String getSuitString() { return suit.toString().toLowerCase(); }

    String getValueString() { return value.toString().toLowerCase(); }
```

Public class with private suits/values which are accessible through getters.

I & T - IT2

```
public abstract class Kaiju implements Damageable, CanAttack {

    String name;
    int healthValue;
    int attackValue;

    public Kaiju(String name, int healthValue, int attackValue) {
        this.name = name;
        this.healthValue = healthValue;
        this.attackValue = attackValue;
    }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public int getHealthValue() { return healthValue; }

    public void setHealthValue(int healthValue) { this.healthValue = healthValue; }

    public int getAttackValue() { return attackValue; }

    public void setAttackValue(int attackValue) { this.attackValue = attackValue; }

    public String roar() { return "Roar!!"; }
```

```
public class KumaDesu extends Kaiju{

    public KumaDesu(String name, int healthValue, int attackValue){
        super(name, healthValue, attackValue);
    }

    public String roar() { return "*Roars in bear*"; }

}
```

```java
public class SaamonDesu extends Kaiju{

    public SaamonDesu(String name, int healthValue, int attackValue){
        super(name, healthValue, attackValue);
    }

    public String roar() { return "Roars in salmon!!"; }

}
```

```java
public class KumaDesuTest {

    @Test
    public void testCanRoar(){
        KumaDesu kumadesu = new KumaDesu("Boris", 500, 70);
        assertEquals( "*Roars in bear*", kumadesu.roar() );
    }

    @Test
    public void testHasHealth(){
        KumaDesu kumadesu = new KumaDesu("Boris", 500, 70);
        assertEquals( 500, kumadesu.getHealthValue() );
    }
```

I & T - IT3

```ruby
def self.all
  sql = 'SELECT * FROM pets;'
  values = []
  pets_hash = SqlRunner.run(sql, values)
  pets = pets_hash.map { |pet| Pet.new(pet) }
  return pets
end
```

```
[➜  db git:(master) ✗ ruby seeds.rb
[1] pry(main)> Pet.all
 => [#<Pet:0x007fdc85a77f38
   @admission_date="2017-07-15",
   @adoptable="Yes",
   @breed="dog",
   @id=159,
   @name="Douglas McKenzie",
   @picture="/douglas_mckenzie.jpg">,
  #<Pet:0x007fdc85a77d08
   @admission_date="2017-06-22",
   @adoptable="No",
   @breed="cat",
   @id=160,
   @name="Gertrude",
   @picture="/gertrude.png">,
```

I & T - IT4

```ruby
    end

    def self.breed_sort
      return Pet.all.sort_by{ |pet| pet.breed }
    end

```

```
[➜  db git:(master) ✗ ruby seeds.rb
[1] pry(main)> Pet.breed_sort
 => [#<Pet:0x007feebd8cd288
   @admission_date="2017-05-25",
   @adoptable="Yes",
   @breed="bird",
   @id=181,
   @name="Terry",
   @picture="/terry.jpg">,
  #<Pet:0x007feebd8cddc8
   @admission_date="2017-08-12",
   @adoptable="Yes",
   @breed="bird",
   @id=176,
   @name="Birdy McBirdface",
   @picture="/birdy_mcbirdface.jpg">,
  #<Pet:0x007feebd8ce200
   @admission_date="2017-06-22",
   @adoptable="No",
   @breed="cat",
   @id=173,
   @name="Gertrude",
   @picture="/gertrude.png">,
  #<Pet:0x007feebd8ce0c0
   @admission_date="2017-07-25",
   @adoptable="Yes",
   @breed="cat",
   @id=174,
   @name="Prudence",
   @picture="/prudence.jpg">,
```

I & T - IT5

```ruby
require("./IT5_array_spec")

class Playlist < Minitest::Test

  attr_reader(:song_name)

  def initialize(song_name)
    @song_name = song_name
  end

end
```

```ruby
require("minitest/autorun")
require("minitest/rg")
require_relative("./IT5_array.rb")

class TestPlaylist < Minitest::Test

  def setup
    @playlist = Playlist.new(["Metallica", "Thin Lizzy", "Céline Dion", "AC/DC"])
  end

  def test_second_artist
    assert_equal("Thin Lizzy", @playlist.second_song)
  end

end
```

```
Random Shizzle — user@CODECLAN015 — ..andom Shizzle — -zsh — 80×
➜  Random Shizzle ruby IT5_array_spec.rb
Run options: --seed 11747

# Running:

.

Finished in 0.001032s, 968.9927 runs/s, 968.9927 assertions/s.

1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
➜  Random Shizzle
```

I & T - IT6

testing_task_1.rb

```ruby
require( "miniTest/autorun" )
require_relative( "./library" )

class TestLibrary < MiniTest::Test

  def setup
    @library = Library.new( [

      { title: "lord of the rings",
        rental_details: {
          student_name: "Jeff",
          date: "01/12/16"
        }
      },
      { title: "lord of the flies",
        rental_details: {
          student_name: "Nicky",
          date: "02/12/16"
        }
      },
      { title: "1984",
        rental_details: {
          student_name: "Thomas",
          date: "03/12/16"
        }
      }
    ] )
  end
```

testing_task_1.rb | testing_task_2.rb

```ruby
class Library

attr_reader(:title, :rental_details, :student_name, :date)
attr_writer(:title, :rental_details, :student_name, :date)

  def initialize(library)
    @library = library
  end

  def all_books
    return @library
  end

  def title_of_first_book
    return @library[0][:title]
  end

end
```

```ruby
  def test_first_book
    book_title = @library.title_of_first_book
    assert_equal( "lord of the rings", book_title )
  end
```

```
Run options: --seed 29577

# Running:

..

Finished in 0.001009s, 1982.1606 runs/s, 1982.1606 assertions/s.

2 runs, 2 assertions, 0 failures, 0 errors, 0 skips
```

IT - IT7

```java
public class Student {
    private int knowledge;
    private ArrayList<Codeable> languages;

    public Student() {
        knowledge = 0;
        languages = new ArrayList<>();
    }

    public void code(Codeable language) {
        knowledge += language.getKnowledge();
        languages.add(language);
    }

    public int getKnowledge() {
        return knowledge;
    }
}
```

```java
public interface Codeable {
    int getKnowledge();
}
```

```java
public class CSharp implements Codeable {
    private int knowledge;

    public CSharp() {
        knowledge = 500;
    }

    public int getKnowledge() {
        return knowledge;
    }
}
```

```java
public class Java implements Codeable {
    private int knowledge;

    public Java() {
        knowledge = 1000;
    }

    public int getKnowledge() {
        return knowledge;
    }
}
```