

## I &amp; 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 &amp; 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*"; }
}

```

```
public class SaamonDesu extends Kaiju{

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

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

}
```

```
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

```
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

```
8     end
9
10    def self.breed_sort
11        return Pet.all.sort_by{ |pet| pet.breed }
12    end
13
```

```
[→ 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

```
song.rb      song_spec.rb      room.rb      IT5_array.rb
1  require("../IT5_array_spec")
2
3  class Playlist < Minitest::Test
4
5    attr_reader(:song_name)
6
7    def initialize(song_name)
8      @song_name = song_name
9    end
10
11  end
12
```

```
IT5_array_spec.rb      IT5_array.rb
1  require("minitest/autorun")
2  require("minitest/rg")
3  require_relative("../IT5_array.rb")
4
5  class TestPlaylist < Minitest::Test
6
7    def setup
8      @playlist = Playlist.new(["Metallica", "Thin Lizzy", "Céline Dion", "AC/DC"])
9    end
10
11    def test_second_artist
12      assert_equal("Thin Lizzy", @playlist.second_song)
13    end
14
15  end
16
```

```
Random Shizzle — user@CODECLAN015 — ..andom Shizzle — -zsh — 80x
[➔ 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
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
end
```

```
testing_task_1.rb      testing_task_2.rb
1  class Library
2
3  attr_reader(:title, :rental_details, :student_name, :date)
4  attr_writer(:title, :rental_details, :student_name, :date)
5
6  def initialize(library)
7    @library = library
8  end
9
10 def all_books
11   return @library
12 end
13
14 def title_of_first_book
15   return @library[0][:title]
16 end
17
18 end
19
```

```
def test_first_book
  book_title = @library.title_of_first_book
  assert_equal( "lord of the rings", book_title )
7 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

```
6
7  public class Student {
8      private int knowledge;
9
10     public Student() {
11         knowledge = 0;
12     }
13
14     public void code(Codeable language) {
15         knowledge += language.getKnowledge();
16     }
17
18     public int getKnowledge() {
19         return knowledge;
20     }
21 }
22
```

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

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

    public CSharp() {
        knowledge = 500;
    }

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

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

    public Java() {
        knowledge = 1000;
    }

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