Examples of Evidence for the Implementation and Testing Unit

I.T 1 - Example of encapsulation

```ruby
class Person
  attr_reader :name, :age

  def initialize(name, age)
    @name = name
    @age = age
  end

  def description
    return "#{name} is #{age} years old"
  end
end
```

I.T 2 - Example the use of inheritance in a program

```ruby
class Mammal
  def eat
    puts "nom nom nom"
  end
end

class Bird < Mammal
  def speak
    puts "tweet tweet"
  end
end

woody = Bird.new
woody.eat
woody.speak
```

I.T 3 - Example of searching and sorting data

```ruby
def index
  @q = Journey.search(params[:q])
  @journeys = @q.result(distinct: true)
  respond_with(@journeys)

end
```

If you do not have an example from past work, write something that you can show and run. Take a screenshot of the code and one of the result for both Algorithms.

I.T 4 - Example of an array, a function that uses an array and the result

```ruby
class_students = ["Bob", "Fred", "George", "Mary", "Sarah"]

def count_students( array )
  total_students = 0

  for student in array
    total_students += 1
  end

  return total_students.to_s + " students in class"
end

puts count_students(class_students)
```

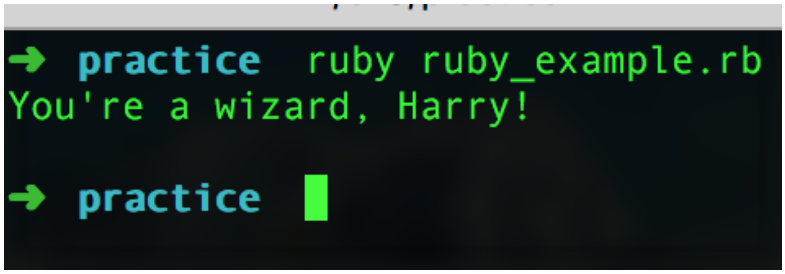
```
➜  practice  ruby ruby_example.rb
5 students in class
➜  practice
```

I.T 5 - Example of a hash, a function that uses a hash and the result

```ruby
wizard1 = { name: "Harry", age: 17, wand_wood: "Elder", wand_length: 15, wand_core: "
  thestral hair", wizard: true}

def wizard_evaluation(my_hash)
  if my_hash[:wizard] == true
    return "You're a wizard, " + my_hash[:name] + "!"
  else
    return "You poor, muggle..."
  end
end

puts wizard_evaluation(wizard1)
```



I.T 6 - Example of polymorphism in a program

```
public class Person {
    private String firstName;
    private String surname;
    private int age;

    Person(String firstName, String surname, int age) {
        this.firstName = firstName;
        this.surname = surname;
        this.age = age;
    }

    // THE POLYMORPHISM BEGINS HERE
    // The point is that we have two methods called setName - one that
    // takes two arguments, and one that takes only one.
    public void setName(String firstName, String surname) {
        this.firstName = firstName;
        this.surname = surname;
    }

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

    // Another example - two methods called setAge that take different types
    // Kind of pointless, but still polymorphism
    public void setAge(int age) {
        this.age = age;
    }

    public void setAge(String age) {
        this.age = Integer.parseInt(age);
    }

    // A third form of polymorphism - dynamic polymorphism - is more of a ballache
    // to do. Basically say we have a class A and a subclass B, which overrides a method
    // called someMethod. If we do the following:
    // B subclass = new B();
    // A superclassRef = subclass;
    // superclassRef.someMethod(); <- this calls B#someMethod *not* A#someMethod.
    // The fact that we call the B method even though we have an A reference is
    // dynamic polymorphism in action.
}
```