Week-2 Day-2

Access Modifiers

- Access modifiers allow us to restrict access to certain properties (fields) or behaviors (methods) in a class from other classes
- We have 4 different access modifiers:
  1. public: fields or methods with this access modifier can be accessible from anywhere inside of our project
  2. protected: fields or methods with this access modifier can be accessible from another class within the same package or a class that is a subclass of that other class (anywhere)
  3. default: same package
  4. private: within the same class only

Encapsulation

- “Wrapping up the data into a single unit”
- Encapsulation is about restricting direct access to variables defined inside of a class and requiring the use of getters/setters to modify the information
- A common misconception is that access modifiers (which can allow us to restrict access) are about “application security”. It is not. Application security is a complicated topic that is not solved by access modifiers
- More fundamentally, having restricted access to variables is more about making sure that developers use the code the way it is meant to be used (so that they cannot change values on an ad-hoc basis)

We take data, and package that data into something nice and tidy

Bach

Tran

24

512-826-2486

What might be a good way to deal with these 4 pieces of data?

Encapsulate the data into an object

```
Class Person {
private String firstName;
private String lastName;
private int age;
private String phoneNumber;

// getters and setters inside here as well
}
```

firstName: Bach
lastName: Tran
age: 24
phoneNumber: 512-826-2486

We can't just make this object out of nowhere, we need a blueprint to actually define what this object looks like