Team Members: Logan Park, Vincent Curran, Kevin Vo

1. What does “design by contract” mean in an OO program? What is the difference between implicit and explicit contracts? Provide a text, pseudo code, or code example of each contract type.

- Design by Contract (DBC) software development technique ensures high-quality software by guaranteeing that every component of a system lives up to its expectations. As a developer using DBC. you specify component contracts as part of the component’s interface. The contract specifies what that component expects of clients and what clients can expect of it. It ensures contracts are made explicit instead of implicit.
-
- `int implicit;`
- `implicit = 4.5;`
- `int explicit;`
- `explicit = (int)4.5;`

- Categorized by the naming conventions, implicit here is going to allow the compiler to automatically convert the double floating number to an int. But in the second example, the explicit specifically states the intention of the programmer by casting the number to int themselves.

Source: https://www.infoworld.com/article/2074956/icontract-design-by-contract-in-java.html
https://www.cloudbees.com/blog/what-is-the-difference-between-implicit-vs-explicit-programming

2. What are three ways modern Java interfaces differ from a standard OO interface design that describes only function signatures and return types to be implemented? Provide a Java code example of each.

- An object-oriented interface (OOI) is the process of designing and creating a user or system interface that is built on object-oriented design interacting objects as the basis of the interface.
- Here are some ways modern Java interfaces differ from these standard OO interface designs.
- Firstly, Java allows default implementations in their interfaces.
    - public interface oldInterface {
    - public void existingMethod();
    - default public void newDefaultMethod() {
    - System.out.println("New default method"
    - " is added in interface");
    - }
    - }
    -
- In addition, they do not allow multiple inheritance (extends), but they allow multiple interface implementations.

Team Members: Logan Park, Vincent Curran, Kevin Vo

    - Public class A implements C,D {...}
- Also, the third of many things Java interfaces differ from a standard OO interface design is that it also has static interface methods.

```
public interface Vehicle {

    // regular / default interface methods

    static int getHorsePower(int rpm, int torque) {
        return (rpm * torque) / 5252;
    }
}
```

Source: https://www.techopedia.com/definition/8640/object-oriented-interface-ooi

https://www.baeldung.com/java-static-default-methods

3. Describe the differences and relationship between abstraction and encapsulation. Provide a Java code example that illustrates the difference.

- Abstraction is the method of hiding the unwanted information or a process of generalization, while encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside, or (more precisely) hide implementation details.
    - One of the things that abstraction is, is that it is a way to describe a particular entity in a reduced way. I can abstract a telephone by its ability to dial and call certain number lines, although telephones can be much more complex in its nature.
- Encapsulation may also be referred to a mechanism of restricting the direct access to some components of an object, such that users cannot access state values for all of the variables of a particular object, Encapsulation can be used to hide both data members and data functions or methods associated with an instantiated class or object.
    - The benefits of Encapsulations include…
        - The ability to hide data, so users have no idea how classes are being implemented or stored. All that the users will know is that values are being passed and initialized.
        - Increasing flexibility. Encapsulation enables the programmer to set variables as read or write-only by using methods like getters and setters.

//This hides the implementation details but we know exactly what we are trying to do.
Encapsulation:
Max(1,2)

//This implies pi can be reduced to just 3.14159 in generality.

Team Members: Logan Park, Vincent Curran, Kevin Vo

Abstraction:
final double PI = 3.14159

4. Class diagram:
https://lucid.app/lucidchart/3fe5be01-af56-47f4-9df1-b432034a95fb/edit?viewport_loc=163%2C69%2C1899%2C1077%2C0_0&invitationId=inv_e2edf275-79cd-4bc2-9d15-55bb3461c7df