Week 5 Research BE – Rachel VanHorn

Prompts

OOP:

1. What are the four pillars of Object-Oriented Programming? Explain each pillar.
2. What is the relationship between a Class and an Object?

Advanced OOP:

3. What are the differences between abstract classes and interfaces? When should you use one over the other?
4. Research the SOLID principles of Object-Oriented Programming (OOP) as introduced by Robert Martin. List the principles, and give a description of each one.

Exceptions:

5. What is an exception?
6. What are the differences between checked and unchecked exceptions?

Testing:

7. What is unit testing and why is it important?

Answers:

1. The four pillars of OOP are

- Abstraction – hiding the complexity of the application from the user. They only see what they need to see.
- Encapsulation – hide information or data that is not necessary for the user to see for security purposes.
- Inheritance – an application can inherit methods and functionality from a parent class or implement methods from multiple interfaces.
- Polymorphism – different objects can handle the same method/function in different ways.

2. The relationship between a Class and an Object is that a Class provides the framework for the Object. A Class works like the blueprint for an Object, so an Object is an instance of the Class that it was created from.

Source: weekly videos

3. An abstract class is a class that cannot be instantiated, only passed down through inheritance to other classes. They can have both abstract and non-abstract methods. So some methods can be defined within the abstract class while the others can be defined in the subclasses. Interfaces are a contract with abstract methods that any class which implements the interface must define and implement. A class can

only inherit from one abstract class, but it can implement multiple interfaces. It would make sense to use an abstract class if the subclasses share many lines of code and if you need access modifiers in your classes other than public. Interfaces are good when you only need public modifiers and when you want every class that implements the interface to use all the methods listed in the interface.

Source:

https://www.geeksforgeeks.org/difference-between-abstract-class-and-interface-in-java/

5/6. An exception is something that disrupts or breaks the flow of a program's normal execution. It is an error that can break your code if you do not specify how to handle its possibility. There are two types of exceptions: checked and unchecked. A checked exception is one that you have to explicitly check for in your code, you do not have control over them; an example would be waiting for data from an API or outside source and having it time out. You prepare for these types of exceptions using try/catch/finally blocks. Inside of the block you tell the program what to try, to catch the exception that may occur and what to do with it and then finally what to do to move forward. An unchecked exception is one that you cannot check for in your code and it is due to programmer error. To prevent these errors, we have to write clean code and use best practices.

Source:

https://docs.oracle.com/javase/tutorial/essential/exceptions/definition.html#:~:text=Definition%3A%20An%20exception%20is%20an,off%20to%20the%20runtime%20system.