Object-Oriented Programming (OOP)

Q1. Why would you create an abstract class, if it can have no real instances?

- [x] to have common behavior in derived classes
- [ ] to explore a hypothetical class
- [ ] to prevent unwanted method implementation
- [ ] to reserve memory for an unspecified class type

Q2. What is the best reason to use a design pattern?

- [x] It will result in code that is more extensible and maintainable
- [ ] It will result in a more compact product.
- [ ] It will speed initial development.
- [ ] It will allow you to add that design pattern to your resume.

Q3. What is encapsulation?

- [ ] defining classes by focusing on what is important for a purpose
- [x] hiding the data and implementation details within a class
- [ ] making all methods private
- [ ] using words to define classes

Q4. What is an IS-A relationship?

- [ ] It implies encapsulation.
- [ ] A superclass object has an IS-A relationship with its subclass.
- [ ] It implies a virtual method.
- [x] A subclass object has an IS-A relationship with its superclass or interface

Q5. You want a method with behavior similar to a virtual method–it is meant to be overridden –expect that it does not have a method body. It just has a method signature. What kind of method should you use?

- [x] an abstract method
- [ ] a public internal method
- [ ] an internal method
- [ ] a protected internal method

Q6. Which type of constructor cannot have a return type?

- [ ] default
- [ ] copy
- [ ] parameterized
- [x] Constructors do not have a return type

Q7 When is a constructor executed?

- [x] when an object is created from a class
- [ ] when an class is defined using the class keyword
- [ ] every time an object is referenced
- [ ] when an object is created from a class using the create keyword

Q8. Which statement best describes the method of inheritance in OOP?

- [x] Inheritance describes the ability to create new classes based on an existing class.
- [ ] Inheritance means that a group of related properties, methods, and other members are treated as a single unit or object.
- [ ] Inheritance forces a class to have a single responsibility from only one parent.
- [ ] Inheritance means that you will never have multiple classes that can be used interchangeably, even though each class implements the same properties or methods in different ways.

Q9. What are the five Creational Design patterns by the Gang of Four ?

- [ ] Observer, State, Strategy, Template Method, and Visitor.
- [ ] Composite, Visitor, State, Prototype, and Singleton.
- [ ] Composite, Builder, Factory Method, Prototype, and Singleton.
- [x] Abstract Factory, Builder, Factory Method, Prototype, and Singleton.

Q10. What is a method?

- [ ] a set of instructions designed to perform a frequently used operation within a program and return no values
- [x] the exact same thing as a function and subroutine
- [ ] a set of variables that can change over time
- [ ] a procedure associated with data and behaviour

Q11. When and how often is a static constructor called?

- [ ] It is called initially when an object is created and called with every new object instance.

- [ ] It is called when an object is destroyed and only one time.
- [x] It is called initially when an object is created and only one time.
- [ ] It is created at time when the object is discarded.

Q12. What does the code shown below demonstrate, and why?

```
static void Multiply(int num1, int num2) {};
static void Multiply(double num1, double num2, double num3) {};
static void Multiply(float num1, float num2) {};
```

- [ ] polymorphism, because each method can perform different task
- [ ] method overriding, because it display the same method name, different or same parameters, and same return type
- [x] method overloading, because it allows the creation of several methods with the same name, wich differ by the type of input via parameter
- [ ] method overriding, because it display the same method name, different parameters, and same return type

Q13. What is the result of using more abstraction?

- [ ] it can increase code vulnerability
- [ ] it can make code unsafe
- [x] it can limit code readability
- [ ] it can be safer for coding

Q14. Which is false for a member function of a class?

- [ ] Member functions can be defined only inside or outside the class body.
- [ ] Member functions can be made to be friends of another class.
- [x] Member functions do not need to be declared inside the class definition.
- [ ] All member functions need to be defined.

Q15. Why is inheritance used when creating a new class?

- [ ] to protect attributes from unwanted changes
- [ ] to delegate coding responsibility more efficiently
- [ ] to conserve memory
- [x] to separate class behavior from the more general one

Q16. In addition to attributes and behaviours, what quality must a class possess?

- [x] a name
- [ ] a state
- [ ] a color
- [ ] an object

Q17. Which type of function among the following shows polymorphism?

- [ ] inline function
- [ ] undefined function
- [x] virtual function
- [ ] class member function

Q18. What best describes what object-oriented programming does?

- [x] It focuses on objects that interact cleanly with one another.
- [ ] It programs exclusively to interfaces.
- [ ] It programs exclusively to classes.
- [ ] It creates one class for all business logic.

Q19. What type of inheritance may lead to the diamond problem?

- [ ] single level
- [ ] multilevel
- [ ] hierarchical
- [x] multiple

Q20. What is the relationship between abstraction and encapsulation?

- [x] Abstraction is about making relevant information visible, while encapsulation enables a programmer to implement the desired level of abstraction.
- [ ] Abstraction and encapsulation are essentially the same.
- [ ] Abstraction and encapsulation are unrelated.
- [ ] Encapsulation is about making relevant information visible, while abstraction enables a programmer to implement the desired level of encapsulation.

Q21. Which of these keywords are access specifiers?

- [ ] abstract and public
- [x] public and private
- [ ] this and final

- [ ] final and abstract

Q22. Why is unit testing harder in OOP than functional programming?

- [x] Objects may maintain internal state, which is not easily accessible by the tests.
- [ ] The quality of unit testing frameworks for functional languages is better.
- [ ] OOP promotes code reuse, which means that your tests have to consider more use cases.
- [ ] Object-oriented languages tend to rely on frameworks such as Spring or Hibernate, which make them difficult to test.

Q23. The open/closed principle states that classes should be open for _ but closed for _.

- [ ] refactoring; duplication
- [ ] modification; duplication
- [x] extension; modification
- [ ] reuse; encapsulation

Q24. Why would you override a method of a base class?

- [ ] to define a method that must be implemented in a derived class
- [x] to define a custom implementation of an inherited member
- [ ] to define a method that must be implemented in a superclass only
- [ ] to define a class that can be inherited from

Q25. There are five classes. Class E is derived from class D, D from C, C from B, and B from A. Which class constructor(s) will be called first if the object of E or D is created?

- [x] A
- [ ] B
- [ ] C
- [ ] C and B

Q26. You have modules that are dependent on each other. If you change one module, you have to make changes in the dependent modules. What term is used to describe this problem, and what is a potential solution?

- [ ] Cohesion. A solution is to show that each module has certain responsibilities and to use an anticohesive design pattern.
- [ ] Encapsulation. A solution is to implement one of the SOLID principles to ensure the modules do not encapsulate with each other.
- [x] Coupling. A solution is to refactor the code to be loosely coupled by using inversion of control and dependency injection.
- [ ] Dependency. A solution is to implement polymorphism and abstraction to change and extract dependent elements of a module so that it functions on its own.

Q27. Which choice is a benefit of using dependency injection?

- [x] loose coupling
- [ ] code reusability
- [ ] lazy initialization
- [ ] data abstraction

Q28. What is the best example of a superclass and subclass relationship?

- [x] car:toyota
- [ ] ducks:pond
- [ ] toes:feet
- [ ] rock:stone

Q29. Which words in the following list are candidates for objects: trumpet, clean, enrage, leaf, tree, collapse, active, and lively?

- [ ] leaf and tree
- [ ] clean, enrage, and collapse
- [ ] clean, active, and lively
- [x] leaf, tree, and trumpet

Q30. How do object behaviour and attributes differ?

- [ ] Behaviour describe dynamic properties; attributes are static.
- [x] Attributes describe a state; behaviours describe a change.
- [ ] Attributes apply only to a specified object; behaviour apply to other linked objects.
- [ ] Behaviours are vector quantities; attributes are scalars.