**OOP’s(Object oriented programming ):**

**Method Overriding Rules**

1. **Only inherited methods can be overridden.**
2. **Final and static methods cannot be overridden.**
3. **The method signature must be the same:**
   * The overriding method must have the same argument list.
4. **Return Type:**
   * The overriding method must have the same return type (or a subtype, known as co-variant return type if the parent class is Object).
5. **Access Modifier:**
   * The overriding method must not have a more restrictive access modifier than the method in the parent class.
6. **Exceptions:**
   * The overriding method must not throw new or broader checked exceptions.
   * If the parent class method throws an exception, the child class method does not need to throw an exception, but if the child class method throws an exception, the parent class method must throw that exception.
7. **Super Keyword:**
   * Use the super keyword to invoke the overridden method from a subclass.
8. **Constructors cannot be overridden.**
9. **Abstract Methods:**
   * Abstract methods must be overridden by the first concrete (non-abstract) subclass.
10. **Static Methods:**
    * A static method in a subclass may hide another static one in a superclass, known as method hiding.
11. **Modifiers:**
    * The synchronized modifier has no effect on the rules of overriding.
    * The strictfp modifier has no effect on the rules of overriding.

**Method Overloading Rules**

1. **Different Method Signatures:**
   * The method signature must not be the same; it must have a different argument list (quantity and types).
2. **Return Types:**
   * May have different return types.
3. **Access Modifiers:**
   * May have different access modifiers.
4. **Exceptions:**
   * May throw different exceptions.
5. **Compile-Time Decision:**
   * Which overloaded method is to be invoked is decided at compile time, based on the actual number of arguments and the compile-time types of the arguments.

**Constructor Rules**

1. **Constructor Name:**
   * The constructor name must be the same as the class name.
2. **Return Type:**
   * Constructors should not have any return type, not even void.
3. **Constructor Overloading:**
   * Constructors can be overloaded.
4. **Default Constructor:**
   * Declaring constructors is not required; if none is supplied, the Java compiler automatically generates a default constructor, which is empty and has no parameters.
5. **Existing Constructor:**
   * The compiler won’t generate a default constructor if there is already a constructor in the class.
6. **Compiler-Generated:**
   * The default constructor is only generated by the compiler.
7. **Inheritance:**
   * Constructors are not inherited.
8. **Access Modifiers:**
   * Constructors can be private, <default>, protected, and public. Other modifiers cannot be used for constructors.
9. **Default Constructor Access:**
   * The default constructor has the same access modifier as the class.
10. **Superclass Constructor Call:**
    * A constructor calls the default constructor of its superclass.
11. **Constructor Chaining:**
    * The first statement in a constructor must call this() or super(), which is called constructor chaining.