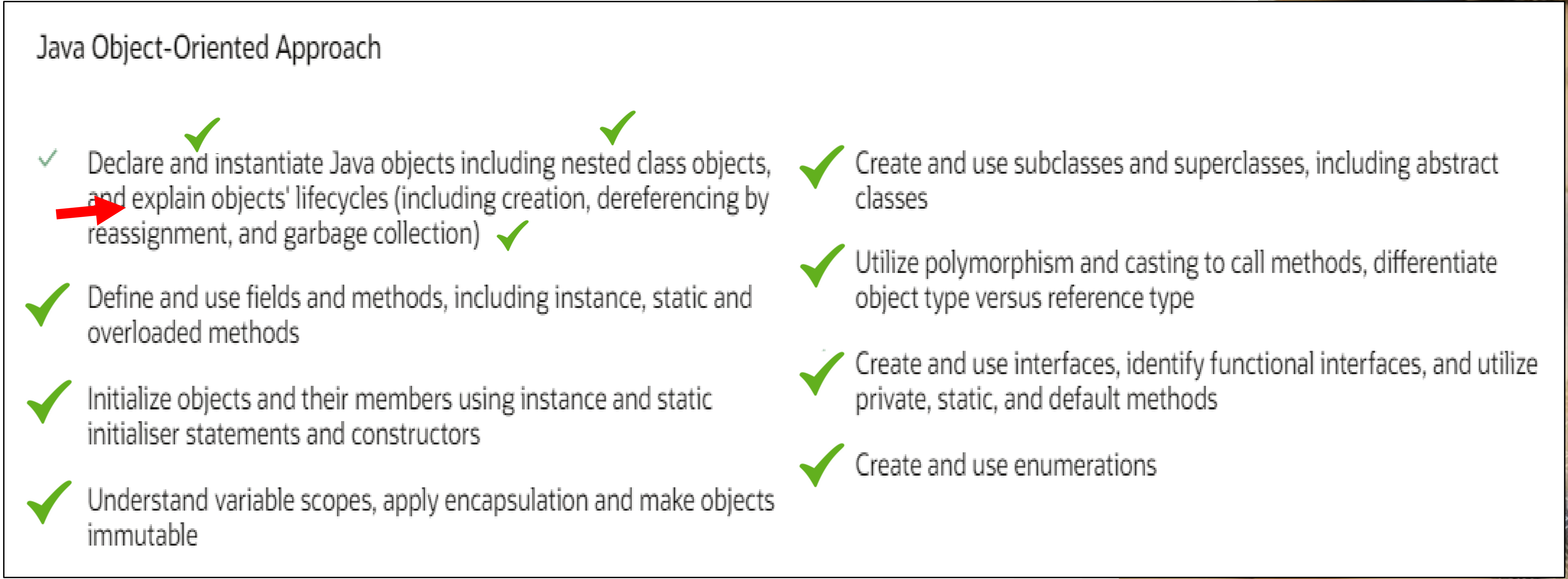
Java Object Oriented Approach

Objects lifecycle

Java Object-Oriented Approach

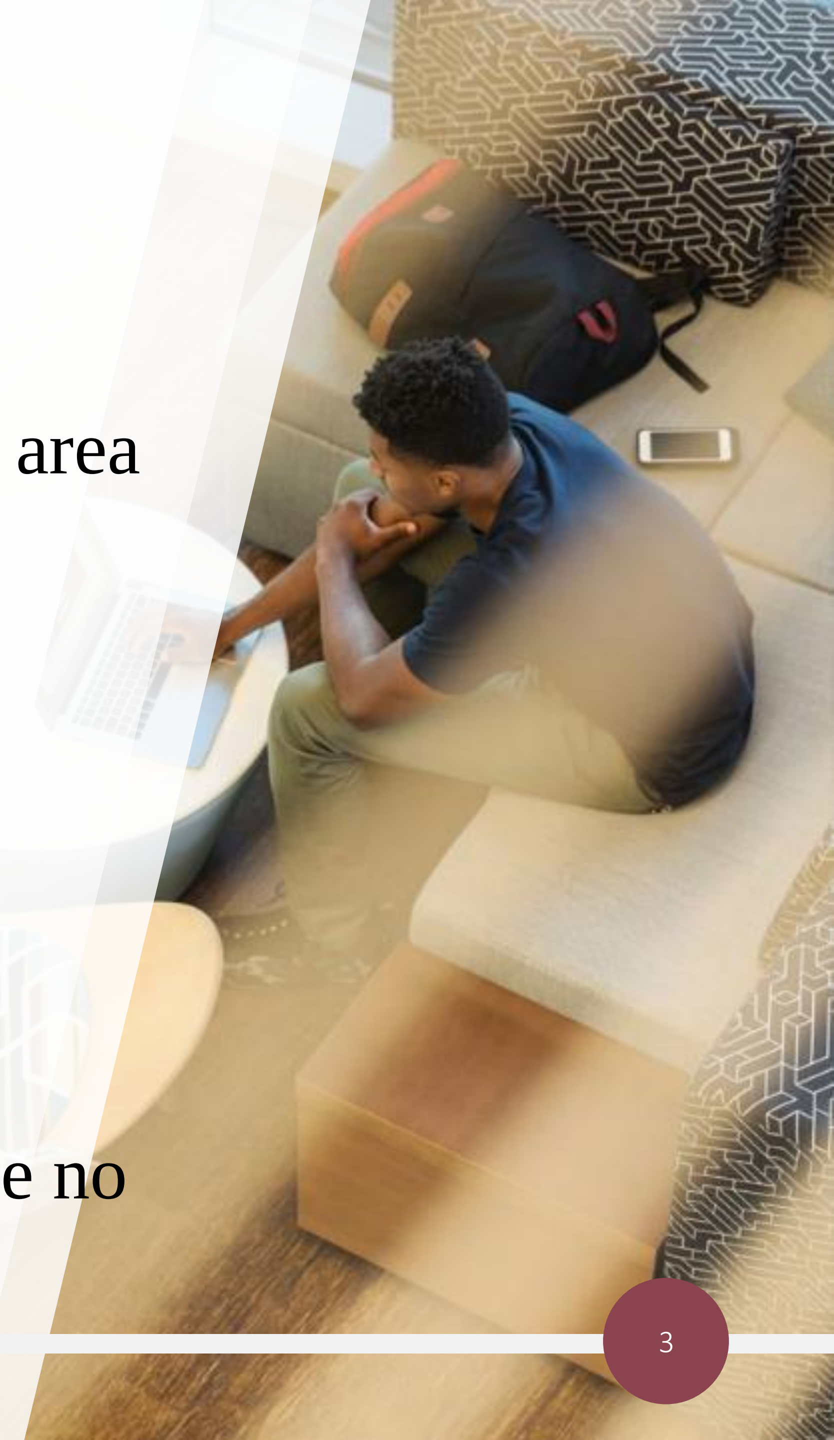


Java Object-Oriented Approach

- ✓ Declare and instantiate Java objects including nested class objects, and explain objects' lifecycles (including creation, dereferencing by reassignment, and garbage collection) ✓
- ✓ Define and use fields and methods, including instance, static and overloaded methods
- ✓ Initialize objects and their members using instance and static initialiser statements and constructors
- ✓ Understand variable scopes, apply encapsulation and make objects immutable
- ✓ Create and use subclasses and superclasses, including abstract classes
- ✓ Utilize polymorphism and casting to call methods, differentiate object type versus reference type
- ✓ Create and use interfaces, identify functional interfaces, and utilize private, static, and default methods
- ✓ Create and use enumerations

Objects Lifecycle

- Local variables are kept on the stack (speed).
- Instance variables and objects live on the heap (large area of memory).
- An object is created using the *new* keyword.
- This allocates storage on the heap for the object.
- The garbage collector is responsible for reclaiming memory no longer needed e.g. objects where there are no longer any references referring to them.



Objects Lifecycle

- When the garbage collector runs is outside our control i.e. the JVM decides. For example, when the system is idle is one possible opportunity.

- *System.gc()* “suggests” that the JVM….