**Functional interfaces and lambdas**

**What is a functional interface?**

* An **interface with exactly one abstract method**.
* Can have **default** and **static methods**.
* Annotated with **@FunctionalInterface** (optional but recommended). If we add the annotation then it will not allow us to add more than one abstract method =>

**Multiple non-overriding abstract methods found in interface org. practice. oops. LivingThing**

A screenshot of a computer code

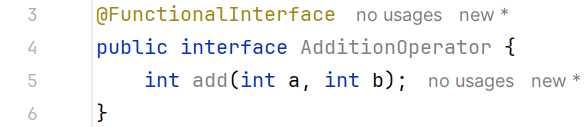
AI-generated content may be incorrect.

**What is lambda expression? And how to use a functional interface with lambda expression?**

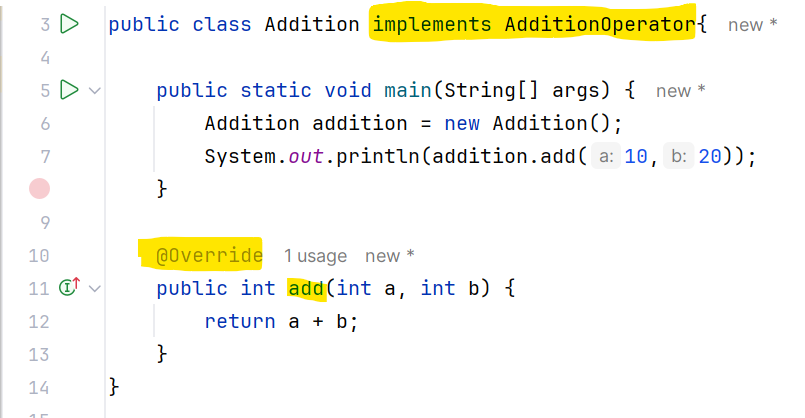
* A short way to write anonymous methods (functional code).
* Used to implement the abstract method of a functional interface.

There are 3 ways to implement a functional interface =>

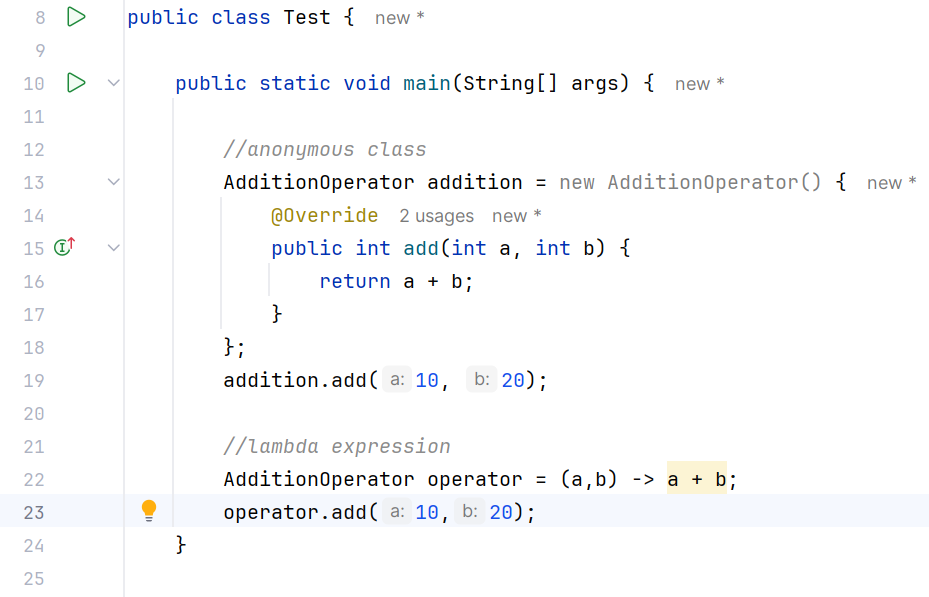
Consider the below functional interface –



1. Using class =>



1. Using anonymous class and lambda expression =>

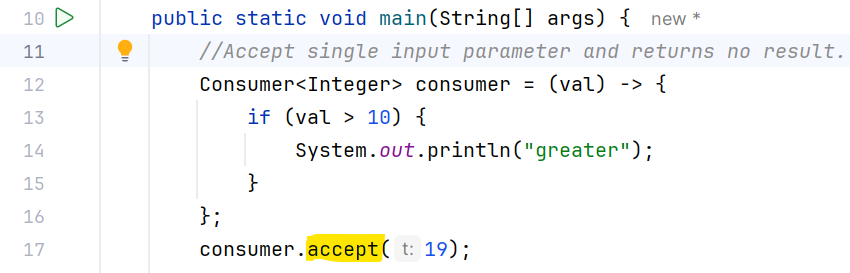


**Advantages of functional interface.**

* Helps in writing **cleaner and concise code**.
* Supports **functional programming** in Java.
* Useful in **streams, APIs**, and **event handling**.
* Easy to pass **behavior as a parameter** (like a method).

**Types of functional interfaces => Consumer, Supplier, Function, Predicate**

**Consumer:**

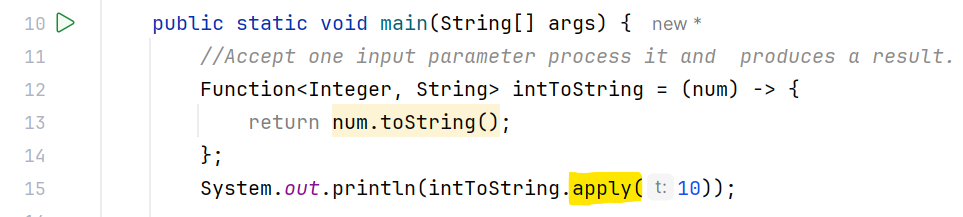
****

**Supplier:**

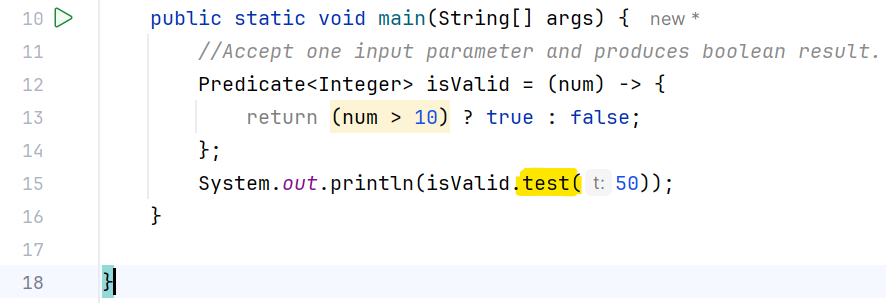
**A screen shot of a computer code

AI-generated content may be incorrect.**

**Function:**

****

**Predicate:**

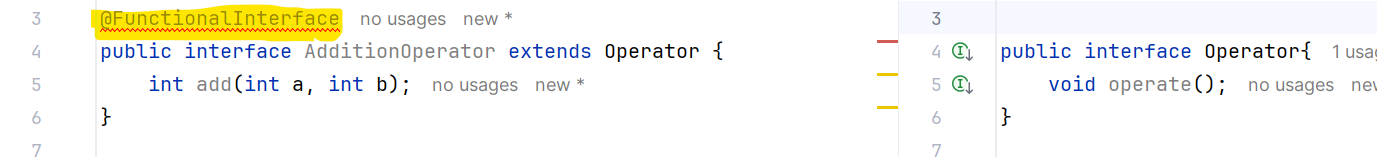
****

**How to handle use case when functional interface extends from another interface (or functional interface)?**

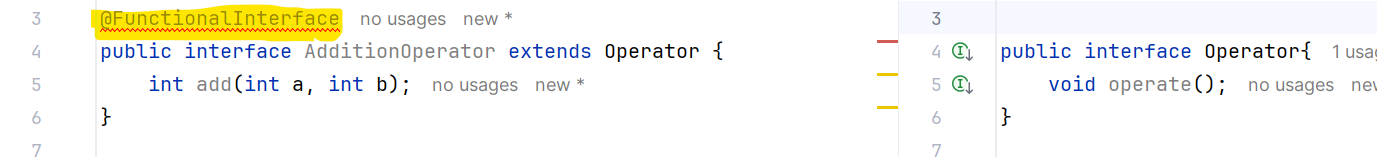
1. **Functional interface extending non-functional interface:**

It must have only one abstract method (either in parent or own) otherwise we will get below error:

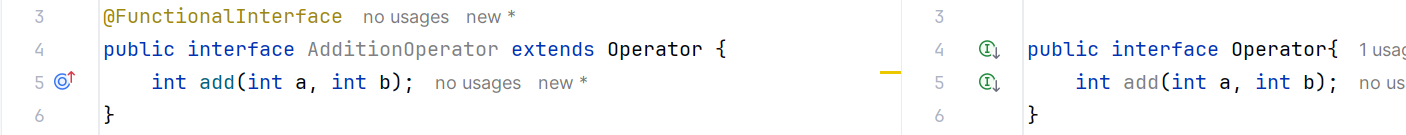
**“Multiple non-overriding abstract methods found in interface org. practice. oops. AdditionOperator”**

****

This will work =>



Also, this will work (same abstract method in parent and child) =>



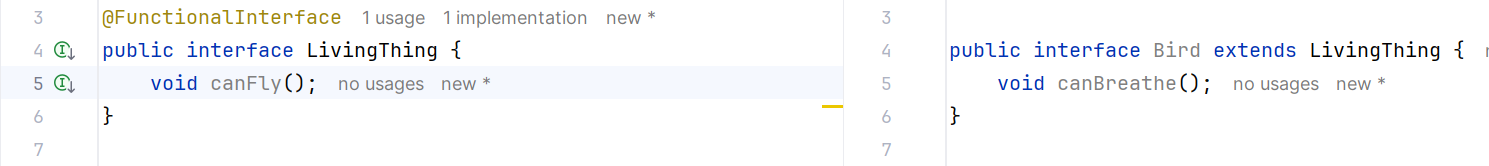
1. **Functional interface extending another functional interface:**

Both must have the same abstract method but can have different static/default methods =>

A white background with black and blue text

AI-generated content may be incorrect.

1. **Interface extending functional interface:**



This works because even though Bird is extending LivingThing, LivingThing still has only one abstract method.