Object-Oriented Programming

University Computer Science · Vocabulary in context

Word	Example
object-oriented programming /ˈɒbdʒekt ˈɔːrɪəntɪd ˈprəʊgræmɪŋ/	In **object-oriented programming**, each book is an object with its own data and methods.
class /klæs/	A **class** is a plan for making objects.
object /ˈɒbdʒekt/	Each car in the game is an **object**.
instance /ˈɪnstəns/	Each dog object is an **instance** of the Dog class. Dog
instance attributes	Each book has its own title in its **instance attributes**.
Class attributes	**Class attributes** are shared by every object of the class.
reference /ˈrefrəns/	A variable holds a **reference** to an object, not a copy.
aliases	a and b are **aliases**: two names for one list. a b
equality /iːˈkwɒlɪti/	**Equality** (==) checks whether two values are the same. (==)
identity /aɪˈdentɪti/	**Identity** (is) checks whether two names point to the very same object. (is)
invariant /ɪnˈveərɪənt/	The loop **invariant** is true every time the loop runs.
encapsulation /ɪnˌkæpsjʊˈleɪʃn/	**Encapsulation** keeps an object's data private.
inheritance /ɪnˈherɪtəns/	With **inheritance**, a Bus class gets the methods of the Vehicle class. Bus Vehicle

exercises.lol exercises.com.cn

Word	Example
composition /ˌkɒmpəˈzɪʃn/	**Composition** means one object contains another: a car has an engine.
fragile base class	Changing a **fragile base class** can break all the classes built on it.
interface /ˈɪntəfeɪs/	The **interface** lists the methods other code can use.
polymorphism /ˈpɒlɪmɔːfɪzəm/	**Polymorphism** lets one method name work in different classes.
duck typing	With **duck typing**, any object with a quack() method can be used as a duck. quack()
dynamic dispatch	With **dynamic dispatch**, the program picks the method while it is running.
substitution /ˌsʌbstɪˈtjuːʃn/	**Substitution** means a subclass object can be used wherever its parent class is expected.
exceptions	Catch **exceptions**, such as a missing file, so the program does not crash.
unit tests	**Unit tests** check one small part of the program at a time.
coupling	Low **coupling** means classes depend on each other very little.
cohesion /kəʊˈhiːʒn/	High **cohesion** means a class does one clear job.
single responsibility principle	The **single responsibility principle** says each class should have one job.

exercises.lol exercises.com.cn

Word	Example
runtime type	The object's **runtime type** decides which method actually runs.
open/closed	The **open/closed** principle says add new code instead of changing old code.
arrange	First, **arrange** the objects you need for the test.
act /ækt/	Next, **act**: call the method you are testing.
assert	Finally, **assert** that the result is what you expected.