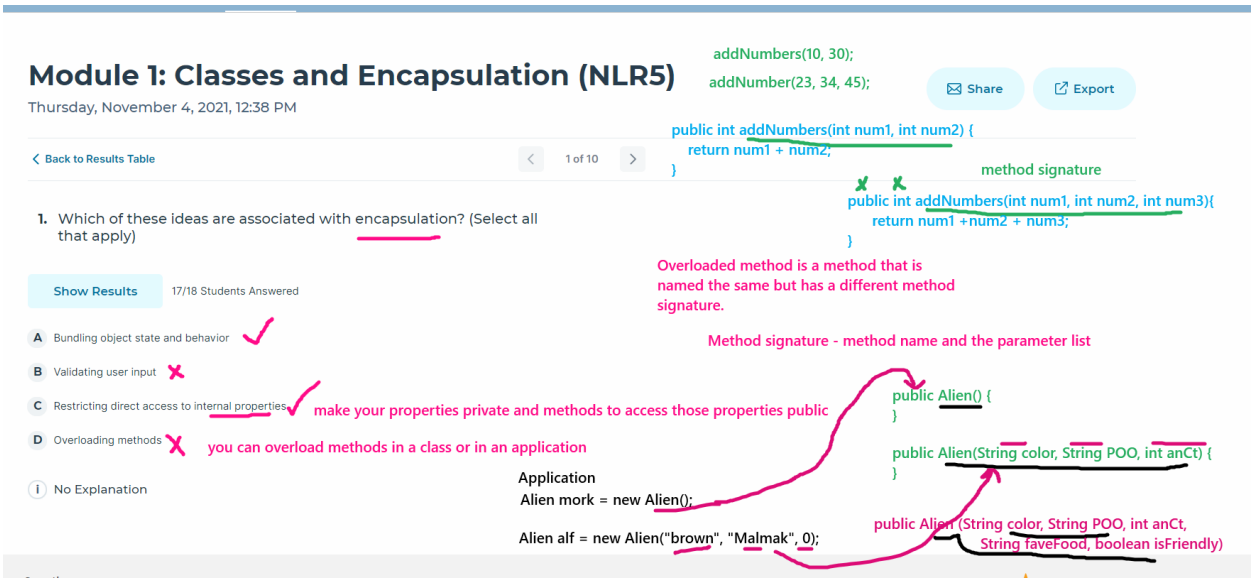
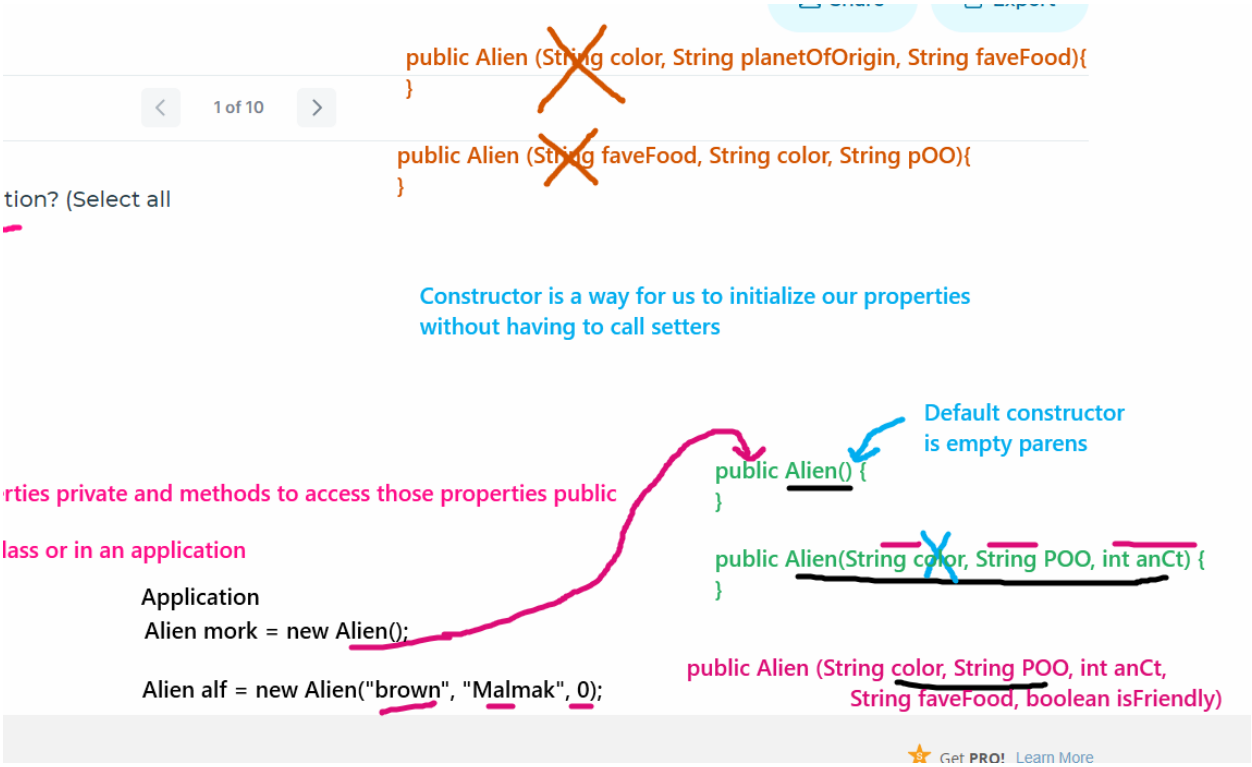
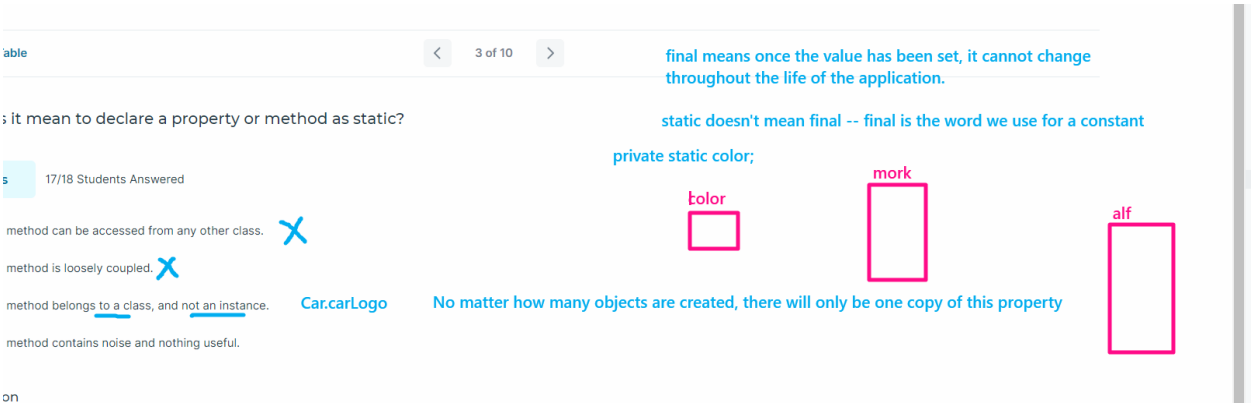
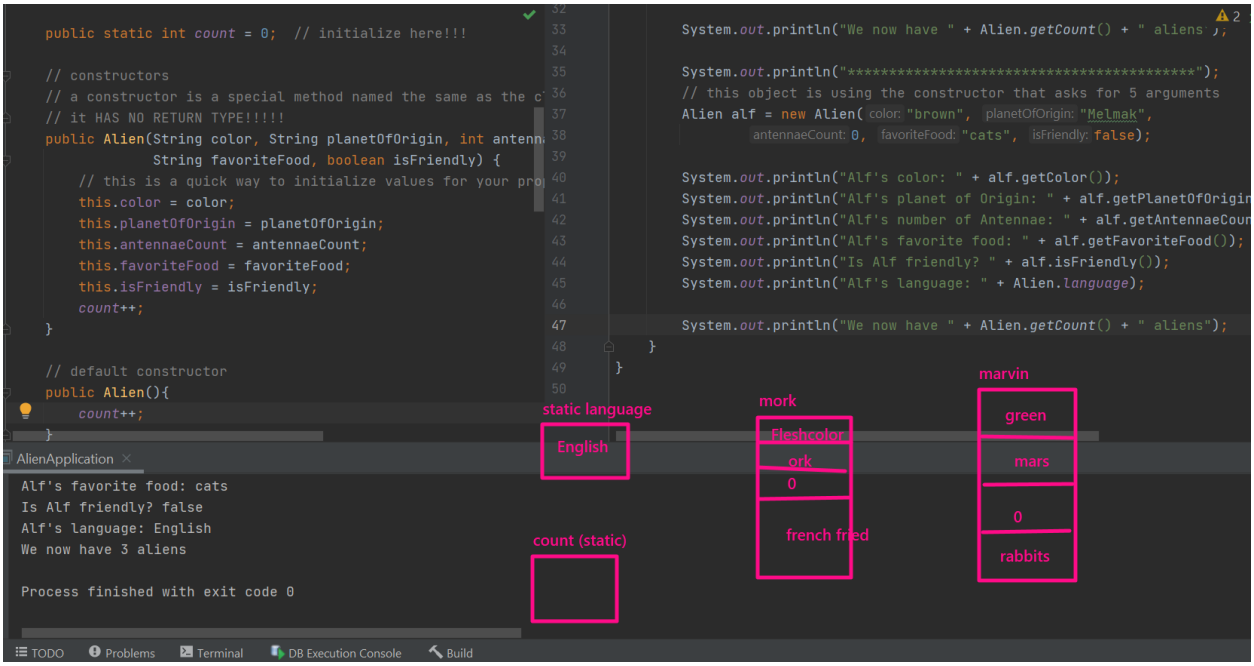

Module 1: Classes and Encapsulation (NLR5)
Thursday, November 4, 2021, 12:38 PM
Share
Export
Back to Results Table
1 of 10
1. Which of these ideas are associated with encapsulation? (Select all that apply)
Show Results
17/18 Students Answered
A Bundling object state and behavior
B Validating user input
C Restricting direct access to internal properties
D Overloading methods
No Explanation
addNumbers(10, 30);
addNumber(23, 34, 45);
public int addNumbers(int num1, int num2) {
return num1 + num2;
}
method signature
public int addNumbers(int num1, int num2, int num3){
return num1 +num2 + num3;
}
Overloaded method is a method that is named the same but has a different method signature.
Method signature - method name and the parameter list
make your properties private and methods to access those properties public
you can overload methods in a class or in an application
Application
Alien mork = new Alien();
Alien alf = new Alien("brown", "Malmak", 0);
public Alien() {
}
public Alien(String color, String POO, int anCt) {
}
public Alien (String color, String POO, int anCt,
String faveFood, boolean isFriendly)


1 of 10
tion? (Select all
public Alien (String color, String planetOfOrigin, String faveFood){
}
public Alien (String faveFood, String color, String pOO){
}
Constructor is a way for us to initialize our properties without having to call setters
rties private and methods to access those properties public
lass or in an application
Application
Alien mork = new Alien();
Alien alf = new Alien("brown", "Malmak", 0);
Default constructor is empty parens
public Alien() {
}
public Alien(String color, String POO, int anCt) {
}
public Alien (String color, String POO, int anCt,
String faveFood, boolean isFriendly)
Get PRO! Learn More


able
3 of 10
final means once the value has been set, it cannot change throughout the life of the application.
it mean to declare a property or method as static?
static doesn't mean final -- final is the word we use for a constant
private static color;
17/18 Students Answered
method can be accessed from any other class.
method is loosely coupled.
method belongs to a class, and not an instance.
method contains noise and nothing useful.
Car.carLogo
No matter how many objects are created, there will only be one copy of this property
color
mork
alf
on


public static int count = 0; // initialize here!!!
// constructors
// a constructor is a special method named the same as the c
// it HAS NO RETURN TYPE!!!!!
public Alien(String color, String planetOfOrigin, int antenna
String favoriteFood, boolean isFriendly) {
// this is a quick way to initialize values for your pro
this.color = color;
this.planetOfOrigin = planetOfOrigin;
this.antennaeCount = antennaeCount;
this.favoriteFood = favoriteFood;
this.isFriendly = isFriendly;
count++;
}
// default constructor
public Alien(){
count++;
}
System.out.println("We now have " + Alien.getCount() + " aliens");
System.out.println("**");
// this object is using the constructor that asks for 5 arguments
Alien alf = new Alien(color: "brown", planetOfOrigin: "Melmak",
antennaeCount: 0, favoriteFood: "cats", isFriendly: false);
System.out.println("Alf's color: " + alf.getColor());
System.out.println("Alf's planet of Origin: " + alf.getPlanetOfOrigin
System.out.println("Alf's number of Antennae: " + alf.getAntennaeCoun
System.out.println("Alf's favorite food: " + alf.getFavoriteFood());
System.out.println("Is Alf friendly? " + alf.isFriendly());
System.out.println("Alf's language: " + Alien.language);
System.out.println("We now have " + Alien.getCount() + " aliens");
}
}
AlienApplication
Alf's favorite food: cats
Is Alf friendly? false
Alf's language: English
We now have 3 aliens
Process finished with exit code 0
TODO
Problems
Terminal
DB Execution Console
Build
static language
English
mork
Fleshcolor
ork
0
french fried
marvin
green
mars
0
rabbits
count (static)

Back to Results Table

4 of 10

4. Which one of the following methods has the same method signature as makeCopies?

```
public String makeCopies(String original, int numberOfCopies) {...}
```

Show Results

17/18 Students Answered

A public String repeatString (String original, int numberOfCopies) {...}

B public String makeCopies(String original, long numberOfCopies) {...}

C public String makeCopies(String original) {...}

D private String makeCopies(String starter, int count) {...}

parameter names don't matter

parameter data types that matter

No Explanation

5. Which of the following statements are correct? (Select all that apply)

Show Results

17/18 Students Answered

A A class is a blueprint that defines the state and behaviors of a data type.

B A class can be overloaded.

Cannot have 2 classes in the same package that have the same method signature

C String is a primitive data type.

D An object is an instance of a class.

No Explanation

Back to Results Table

7 of 10

7. What occurs when the following code co

```
public class Greeter {
    ...
    public String happyBirthday(String name, int age) {
        return "Happy Birthday " + name + "! You are " + age + " years old.";
    }
    public String happyBirthday(int numberOfCandles, String message) {
        return message + "Wow! Your cake has " + numberOfCandles + " candles.";
    }
    ...
}
```

overloaded

Show Results

17/18 Students Answered

A The happyBirthday method is *overloaded* by the Java comp

B The Java compiler throws an error saying there are duplicate happyBirthday methods because they have the same name, and both have int and String parameter types.

C It won't compile since you can't concatenate int values, such as age or numberOfCandles to a string.

D It won't compile since the methods must use *this* to refer to the variables.

this refers to properties at the class level, not local variables

No Explanation

8. Given the following code, which is the correct getter for the derived property fullName?

Show Results 17/18 Students Answered

A
```
public String getFullName(String lastName, String firstName){
return lastName + ", " + firstName;
}
```

B
```
private String fullName; //Additional instance variable

public String getFullName(){
return this.fullName;
}
```

C
```
public String getFullName(){
return this.lastName + ", " this.firstName;
}
```

D
```
public void getFullName(){
String fullName = this.lastName + ", " + this.firstName;
}
```

9. Java packages offer which of the following benefits? (Select all that apply)

Show Results 17/18 Students Answered

A They reduce memory since all classes in a package are compiled at the same time.

B They prevent your type names from colliding with others. your.Employee can be distinguished from their.Employee.

same class name defined in different packages

C They allow you to gather classes to logically relate them together.

D The Java compiler uses packages to optimize compilation which speeds up build-time.

i No Explanation

10. Does Java allow an *instance* method to directly call a static method in the same class like this?

Show Results 17/18 Students Answered

A Yes, any method, regardless of whether it is static or not, can call any other method within the same class.

B It depends on whether the correct access modifiers are used. They need visibility to one another.

C Yes, the method signature rules permit instance methods to call static methods provided there is no reference to *this*.

D Yes, any static methods of a class are available to the instance methods of that class.

i No Explanation