

Java Polymorphism Method Overloading - Assignment

Instructions:

You are provided with the `PolymorphismMethodOverloadingAssignment.java` class. Your task is to implement all the below points in it to demonstrate method overloading and polymorphism. Below are the specific tasks you need to complete, including exact variable names and method calls.

Task 1: Demonstrating Method Overloading

1. **Objective**: Implement method overloading by creating multiple `speak()` methods in the `Animal` class and the `Dog` class.

2. **Details**:

- You need to implement three overloaded `speak()` methods:

- 1. A method that takes no parameters and prints a general message about the animal making a sound.

- 2. A method that accepts a `String` parameter and prints the sound the animal makes.

- 3. A method that accepts an `int` parameter and prints how many times the animal speaks.

3. **Steps**:

- Create an `Animal` class:

- Implement the `speak()` method with no parameters and printing "The animal makes a sound."

- Implement the `speak(String sound)` method to print the sound the animal makes as "The animal says: " + sound.

- Implement the `speak(int times)` method to print how many times the animal speaks as "The animal speaks " + times + " times."

- Create the `Dog` class, which extends `Animal`:

- Override the `speak()` method to print a dog-specific sound as "The dog barks."

- Override the `speak(String sound)` method to print the specific dog sound passed as a parameter as "The dog says: " + sound.

- Override the `speak(int times)` method to print how many times the dog barks as "The dog barks " + times + " times."

Task 2: Implement main()



1. ****Objective****: Implement main() as:

- Create an object of Dog as dog.
- Invoke all 3 methods as:
 - speak()
 - speak("Woof")
 - speak(3)

Final Deliverable:

1. Implement the `main()` method in the `PolymorphismMethodOverloadingAssignment.java` class.
2. Ensure that:
 - You ****create a `Dog` object**** and demonstrate method overloading by calling the `speak()` method in different ways:
 - Call `speak()` with no parameters to print a generic message.
 - Call `speak(String sound)` to print the specific sound the dog makes.
 - Call `speak(int times)` to print how many times the dog barks.
3. Do not include any print statements outside of the method overloading demonstration.

Execution Steps to Follow:

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top)  Terminal  New Terminal.
3. This editor Auto Saves the code.
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.

5. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.

6. To run your project use command:

```
mvn compile exec:java -  
Dexec.mainClass="com.yaksha.assignment.PolymorphismMethodOverloadingAssignment"
```

7. To test your project test cases, use the command

```
mvn test
```

You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.