Fundusze Europejskie Wiedza Edukacja Rozwój
Politechnika Wrocławska
Unia Europejska Europejski Fundusz Społeczny

„ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej”

Programming Paradigms Tutorials
Object Oriented Programming in Java

Polymorphism

Polymorphism refers to the ability of OOPs programming languages to differentiate between entities with the same name efficiently. This is done by Java with the help of the signature and declaration of these entities.
Polymorphism in Java are mainly of 2 types Overloading and Overriding.

Inheritance

Inheritance is an important pillar of OOP(Object Oriented Programming). It is the mechanism in java by which one class is allowed to inherit the features (fields and methods) of another class.

- Super Class: The class whose features are inherited is known as superclass (or a base class or a parent class).
- Sub Class: The class that inherits the other class is known as subclass (or a derived class, extended class, or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.
- Reusability: Inheritance supports the concept of “reusability”, i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

Syntax:
```
class derived-class extends base-class
{
   //methods and fields
}
```

Encapsulation

Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. Another way to think about encapsulation is, it is a protective shield that prevents the data from being accessed by the code outside this shield.

- Technically in encapsulation, the variables or data of a class is hidden from any other class and can be accessed only through any member function of own class in which they are declared.
- As in encapsulation, the data in a class is hidden from other classes, so it is also known as data-hiding.
- Encapsulation can be achieved by Declaring all the variables in the class as private and writing public methods in the class to set and get the values of variables.

Abstraction

Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or the non-essentials units are not displayed to the user. Ex: A car is viewed as a car rather than its individual components.

Data Abstraction may also be defined as the process of identifying only the required characteristics of an object ignoring the irrelevant details. The properties and behaviors of an object differentiate it from other objects of similar type and also help in classifying/grouping the objects.
In java, abstraction is achieved by interfaces and abstract classes. The interfaces are mostly used to define contract with other parts of the application. Abstract class is mostly used to extract parts of similar classes.

Class

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:

- Modifiers: A class can be public or has default access (Refer this for details).
- Class name: The name should begin with a initial letter (capitalized by convention).
- Superclass(if any): The name of the class's parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
- Interfaces(if any): A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
- Body: The class body surrounded by braces, { }.

Object

It is a basic unit of Object Oriented Programming and represents the real life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. An object consists of:

- State : It is represented by attributes of an object. It also reflects the properties of an object.
- Behavior : It is represented by methods of an object. It also reflects the response of an object with other objects.
- Identity : It gives a unique name to an object and enables one object to interact with other objects.

Exercises

1. Define class dog and explain which parts of it are related to identity, state, behavior.

2. Design interface for shape (square, rectangle, triangle, circle).

3. Using interface from ex. 2 design classes for square, rectangle, triangle, circle with implementation. Please remember about inheritance and encapsulation.

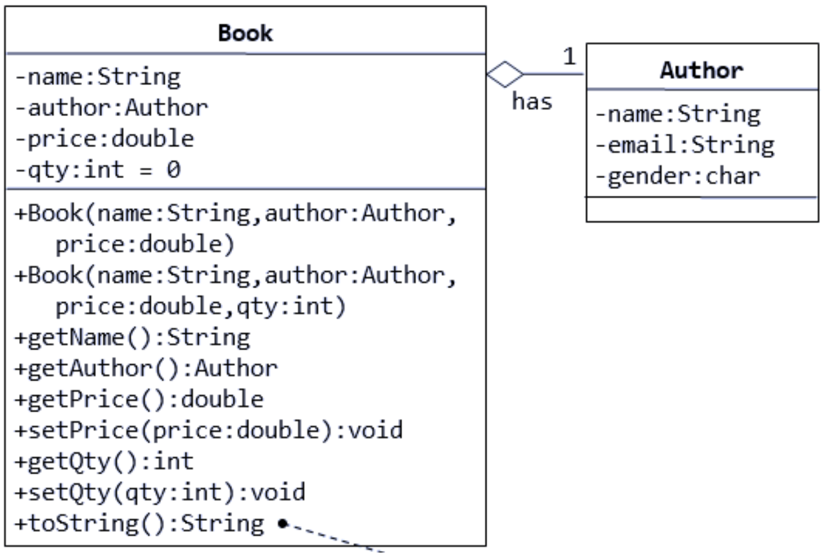
4. Define classes Car, SportCar, SuperCar
   Show how to use abstraction, inheritance, encapsulation and explain your solution.

5. For class User define equals and hashcode

```
public class User {

     private int id;
     private String name;
     private String email;
     private String mobile;
}
```

6. Modify below class(es) to support multiple authors for one book.


Book
-name:String
-author:Author
-price:double
-qty:int = 0
+Book(name:String,author:Author,
   price:double)
+Book(name:String,author:Author,
   price:double,qty:int)
+getName():String
+getAuthor():Author
+getPrice():double
+setPrice(price:double):void
+getQty():int
+setQty(qty:int):void
+toString():String
has
1
Author
-name:String
-email:String
-gender:char