

## Java Fundamentals

### 4-2: Object and Driver Classes

### Practice Activities

#### Lesson Objectives:

- Describe the general form of a Java program
- Describe the difference between an Object class and a Driver class
- Access a minimum of two Java class APIs
- Explain and give examples of Java keywords
- Create an Object class
- Create a Driver class

#### Vocabulary:

Identify the vocabulary word for each definition below.

Packages	A group of related Java classes.
Code blocks	Sections of code that are enclosed inside a set of curly braces. {}
Upper Camel Case	First letter uppercase and the first letter of each internal word capitalized. Example: SavingsAccount
Constant	A named value that does not change.
Lower Camel Case	First letter lowercase and the first letter of each internal word capitalized. Example: studentFirstName
main class	A class that contains a main method.
import statement	A code statement in a Java class file that includes java code from another package or class.
base class	A class that defines instances of objects to be used in another class.
Single line comment	Code that is preceded by //. Comments are used to clarify programming logic. Comments are ignored by the compiler.
Keyword	A word that has a special function in the Java language, and cannot be used as names for classes, methods, or variables.
Java Class Library	The library of Java classes available to import into a programmer-created class.
	The outline of an object, including class variables, constructors, and methods.
	A special kind of method that is a template for an object.
	Values that are sent into a method or constructor to be used in a calculation or substituted with values from the class.
	Values, such as numbers, characters, or booleans. References to objects, such as a BankAccount object.
	Keywords used to specify the accessibility of a class (or type) and its members. Ex: public, private, protected, default
	A block of code inside a class that is used to change or access information about the class.

## Try It/Solve It:

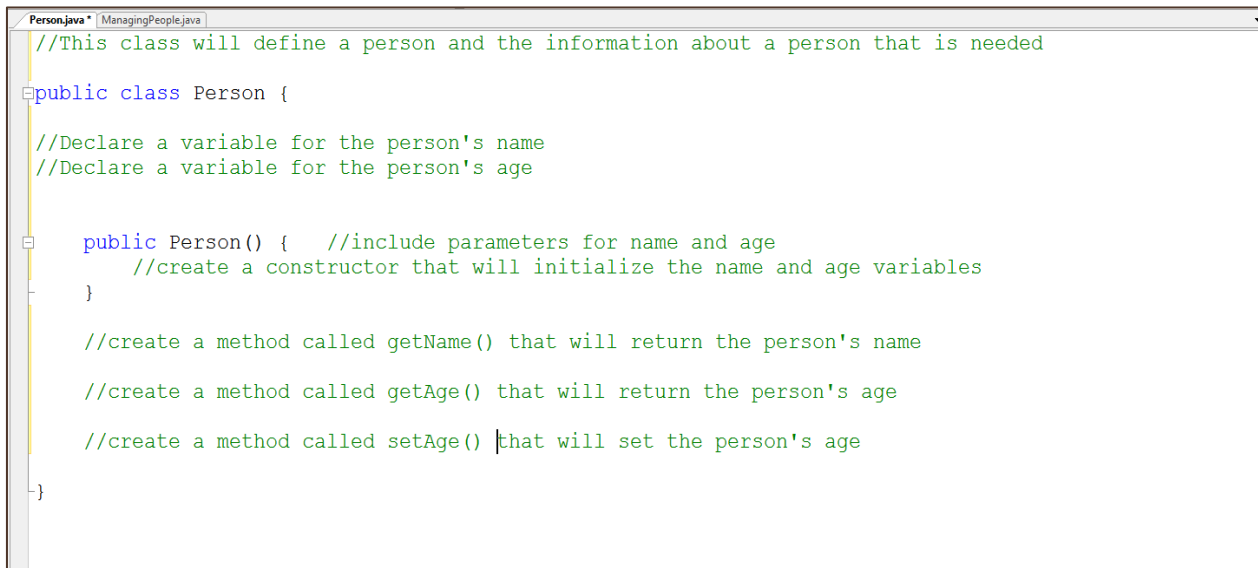
1. Name the components that comprise a .java file. List the components in the order that you would expect to see them in a Java program.   
1. Package Declaration, 2.Import Statements, 3. Class Declaration, 4. Field Declarations, 5. Constructor Declarations, 6. Method Declarations, 7. Inner Class Declarations.
2. Describe the difference between upper camel case and lower camel case and provide an example of when you would them.   
Upper camel case is when first letter of each word is capitalized where as lower camel case is the first letter of the first word is lowercase and the first letter of each next word is capitalized. Ex: ThisIsUpper, thisIsLower
3. What syntax is used to import the entire Java utilities package? And if you import an entire package do you also need to import additional classes in the same package separately?   
A. import java.util.\*;      B: If you import the entire package you do not need to import additional classes.
4. Write the syntax for a simple Java object class named Student with the following format:

Student Name: Lisa Palombo  
Student ID: 123456789  
Student Status: Active

The student information will be stored in the following variables:

fName, lName, stuId, stuStatus.

5. Write the code for a Driver Class that will create a Student Object and print the information about the object to the screen.
6. From this lesson, list 10 Java keywords.   
public, class, static, void, main, String, int, if , else, return
7. Complete the programmer-created object class below. Read the comments for instructions.



```
Person.java * ManagingPeople.java
//This class will define a person and the information about a person that is needed

public class Person {

    //Declare a variable for the person's name
    //Declare a variable for the person's age

    public Person() {    //include parameters for name and age
        //create a constructor that will initialize the name and age variables
    }

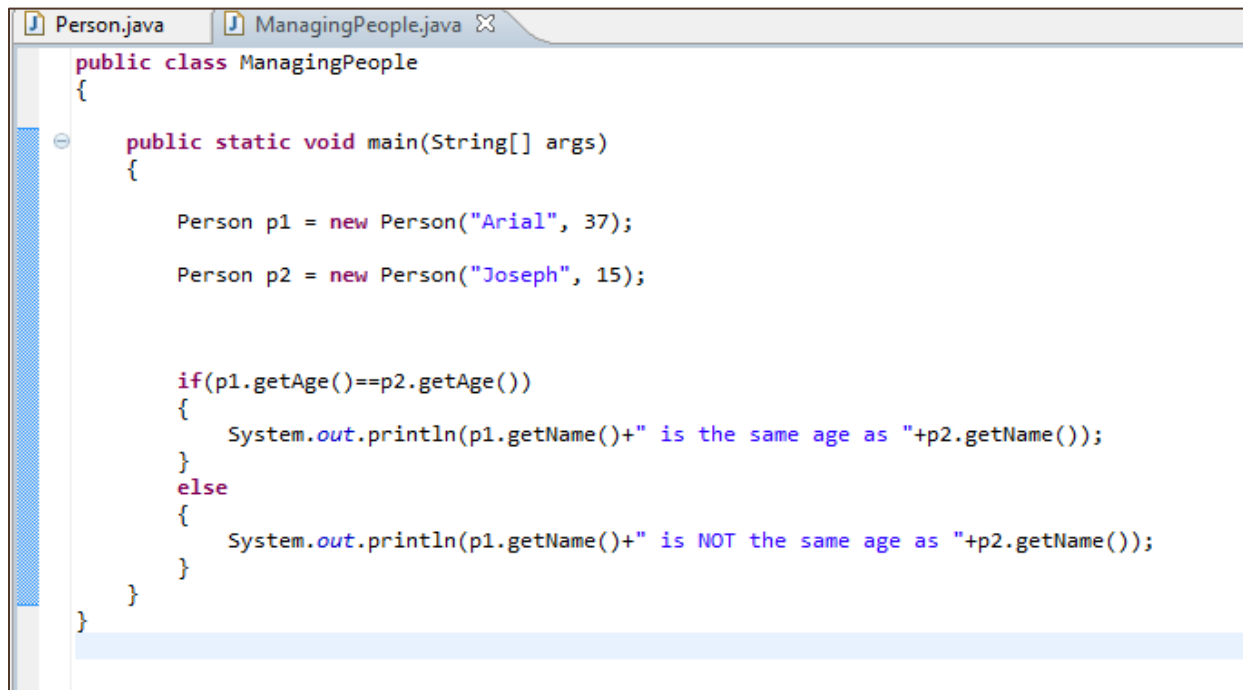
    //create a method called getName() that will return the person's name

    //create a method called getAge() that will return the person's age

    //create a method called setAge() that will set the person's age

}
```

8. Use the following driver class to test your results from above.



```
public class ManagingPeople
{
    public static void main(String[] args)
    {
        Person p1 = new Person("Arial", 37);
        Person p2 = new Person("Joseph", 15);

        if(p1.getAge()==p2.getAge())
        {
            System.out.println(p1.getName()+" is the same age as "+p2.getName());
        }
        else
        {
            System.out.println(p1.getName()+" is NOT the same age as "+p2.getName());
        }
    }
}
```