

# Java Fundamentals

## 4-2: Object and Driver Classes

### Practice Solutions

#### Vocabulary:

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

**package**  
**import**  
**class**  
**variables (or fields)**  
**constructors**  
**methods**

2. Describe the difference between upper camel case and lower camel case and provide an example of when you would them.

**Upper camel case would be used for a class name. Upper camel case begins with an uppercase letter.**  
**Lower camel case would be used for a variable name. Lower camel case begins with a lowercase letter.**

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?

**import java.util.\*;**  
**No you do not need to import additional classes of the same package**

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, stuld, stuStatus.

**Answers will vary. Example:**

```
import java.lang;
public class Student
{
    public String fName;
    public String lName;
    public int stuld;
    public String stuStatus;

    public Student(){
}

    public getfName(){
        return fName;
    }
    ...
}
```

5. Write the code for a Driver Class that will create a Student Object and print the information about the object to the screen.

**Answers will vary. Example:**

```
public class StudentTester
{
    public static void main(String[] args){
        Student s1 = new Student();
        ...
        System.out.println("Student Name: "+fName+" "+lName);
        System.out.println("Student ID: "+stuld);
        System.out.println("Student Status: "+stuStatus);
    }
}
```

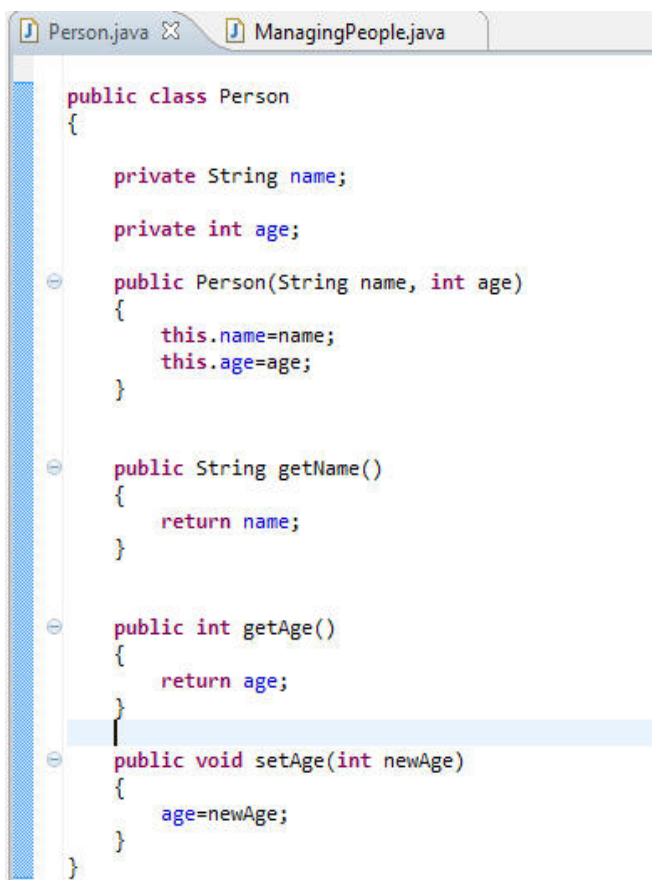
6. From this lesson, list 10 Java keywords.

**Answers will vary but should contain some of these:**

**package, import, public, class, int, void, main, return, static, if, else, for**

7. Complete the programmer-created object class below. Read the comments for instructions.

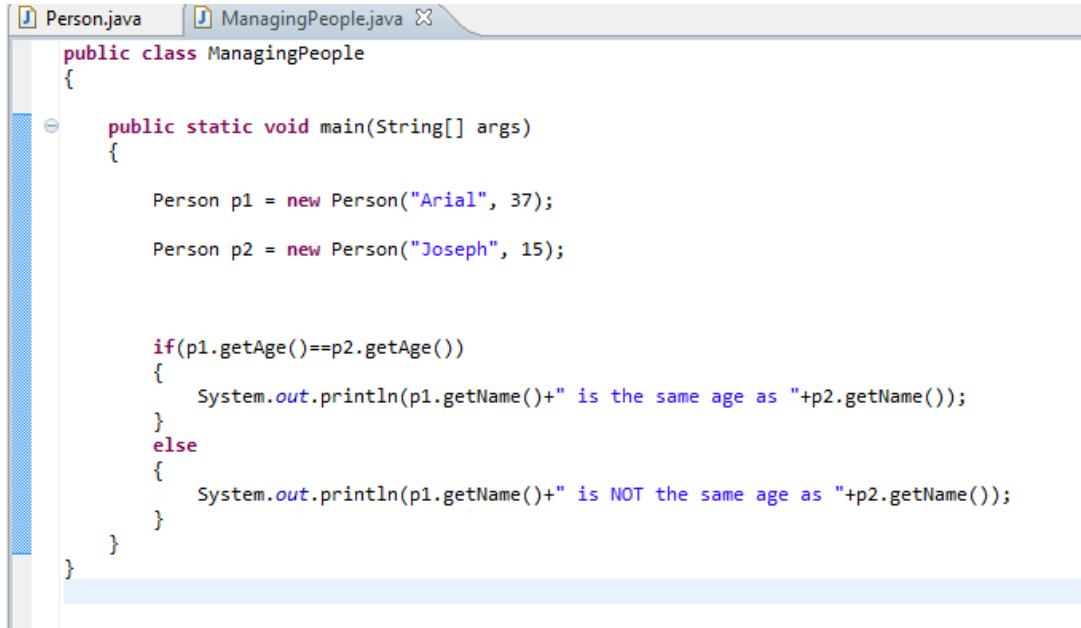
**The answer is displayed in the image below. Comments have been removed.**



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

**Possible modifications to the code could include changing a person's age or adding more people to the comparison.**

**Make sure, when compiling, that both classes are saved in the same package.**



```
Person.java ManagingPeople.java X
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());
        }
    }
}
```