
Part 1: *Create a program with the following requirements:*

Starting with the superclass, create the object Person:

- Person needs to have two private class attributes for name and job.
- Both attributes must be defined as String variables.

Create two subclasses of Person:

- The two subclasses are SuperCitizen and NormalCitizen.
- SuperCitizen and NormalCitizen must have the ability to access the two attributes of the Person superclass.
- The SuperCitizen class also needs three private class attributes – a superPower attribute (assigned as a String), a cape attribute (assigned as a boolean), and a powerLevel attribute (assigned as an int).
- NormalCitizen must also have the ability to access the two attributes of the Person superclass and will have no special attributes.

Finally, you must construct two subclasses of SuperCitizen – SuperHero and SuperVillain:

- SuperHero must have the ability to access the two attributes of the Person superclass as well as all three attributes of the SuperCitizen class.
- SuperHero also needs its own private class attribute called catchPhrase (to be stored as a String).
- SuperVillain, like SuperHero, must access all of the same superclass attributes as SuperHero as well as having its own attribute evilPlan (to be stored as a String).

SCROLL TO NEXT PAGE for more specific requirements for each of your object class files.

Constructor Requirements:

- Person – No default constructor (must not have a default constructor). It must have a 2 parameter constructor with the ability to initialize each of the 2 class variables.

- SuperCitizen & NormalCitizen - No default constructor (must not have a no-parameter constructor). SuperCitizen must have a 5-parameter constructor and the ability to access the two superclass variables as well as initialize the 3 class variables (in SuperCitizen). NormalCitizen must have a 2-parameter constructor and the ability to access the two superclass variables.

- SuperHero & SuperVillain - No default constructor (must not have a no-parameter constructor). Each must have a 6-parameter constructor with the ability to access all 5 superclass variables as well as initialize the class variable.

Method Requirements:

- Person –
  - Superclass
  - Accessor methods that will return the values for the two attributes of the class.
  - toString Sample:
    - name:"Michael Scott",job="Regional Manager"

- SuperCitizen –
  - Subclass of Person
  - Accessor methods for the 3 class variables.
  - A void method called powerLevelModification receives an int value for newLevel and modifies the powerLevel value to the newLevel.
  - toString Sample:
    - name:"Clark Kent",job:"Journalist",superPower: "Invincibility & Superstrength",cape:true,powerLevel:100

- NormalCitizen –
  - Subclass of Person
  - No additional methods are necessary.

- SuperHero –
  - Subclass of SuperCitizen
  - Accessor method for the class variable.
  - Override the powerLevel accessor method to increase power level by a random int value between 1 and 10.
  - toString Sample:

- name:"Clark Kent",job:"Journalist",superPower: "Invincibility & Superstrength,cape:true,powerLevel:10, catchphrase:"Up, up, & away!"

- SuperVillain –
  - Subclass of SuperCitizen
  - Accessor method for the class variable.
  - toString Sample"
    - name:"Lex Luthor",job:"Scientist",superPower:"Genius Level Intellect",cape:false,powerLevel:6,evilPlan:"Use a Kryptonite Ring to defeat Superman"

When you are finished with your program, use the following test code to ensure that all classes, methods, etc. work correctly!

```
 1 Person p1 = new Person("Michael Scott", "Regional Manager");
 2 SuperCitizen s1 = new SuperHero("Clark Kent", "Journalist",
   "Invincibility & Superstrength", true, 10, "Up, up, & away!");
 3 SuperCitizen s2 = new SuperVillain("Lex Luthor", "Scientist", "Genius
   Level Intellect", false, 6, "Defeat Superman with a Kryptonite Ring");
 4 SuperHero s3 = new SuperHero("Peter Parker", "Photographer", "Spider
   Abilities", false, 9, "With great power comes great responsibility");
 5 System.out.println(p1);
 6 System.out.println(p1.getName()+" "+p1.getJob());
 7 System.out.println(s1);
 8 System.out.println(s1.getName()+" "+s1.getJob()+" "+s1.getSuperPower()+"
   "+s1.hasCape()+" "+((SuperHero)s1).getPowerLevel()+"
   "+((SuperHero)s1).getCatchphrase());
 9 s1.powerLevelModification(100);
10 System.out.println(s1);
11 System.out.println(s2);
12 System.out.println(s2.getName()+" "+s2.getJob()+" "+s2.getSuperPower()+"
   "+s2.hasCape()+" "+s2.getPowerLevel()+"
   "+((SuperVillain)s2).getEvilPlan());
13 System.out.println(s3);
14 System.out.println(s3.getName()+" "+s3.getJob()+" "+s3.getSuperPower()+"
   "+s3.hasCape()+" "+s3.getPowerLevel()+" "+s3.getCatchphrase());
```

Check-In Questions (Be ready to share next class):

- **Which lines of code in the test driver apply the concept of polymorphism?**
- **What happens if you take out the casting statement to SuperHero in Line 7 / SuperVillain in Line 11?**
- **Why does the object for Spiderman allow a programmer to call getCatchphrase() without casting s3 as a SuperHero in the statement?**