

# Inheritance: Concept Challenge



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/)  
by Christine Alvarado, Mia Minnes, and Leo Porter, 2015.

# Concept Challenge

- **Pause** Try to solve the problem yourself
- **Discuss** with other learners (if you can)
- **Watch** the UCSD learners video
- **Confirm** your understanding with our explanation



```
public class Person {  
    private String name;  
  
    public Person( String n ) {  
        super();  
        this.name = n;  
    }  
    public void setName( String n ) {  
        this.name = n;  
    }  
}
```

```
Student s = new Student();
```

```
public class Student extends Person {  
    public Student () {  
        this.setName("Student");  
    }  
}
```



# Start IVQ

```
public class Person {  
    private String name;  
  
    public Person( String n ) {  
        super();  
        this.name = n;  
    }  
    public void setName( String n ) {  
        this.name = n;  
    }  
}
```

```
public class Student extends Person {  
    // changed for example  
    public Student () {  
        this.setName("Student");  
    }  
}
```

<IVQ placeholder> (D – no default ctor in Person)

Suppose you call:  
Student s = new Student();

What will be the name variable for this object?

- A. "Student"
- B. "Undefined"
- C. null
- D. Compile Error
- E. Runtime Error



End IVQ / Start Discussion

```
public class Person {  
    private String name;  
  
    public Person( String n ) {  
        super();  
        this.name = n;  
    }  
    public void setName( String n) {  
        this.name = n;  
    }  
}
```

```
Student s = new Student();
```

```
public class Student extends Person {  
    public Student () {  
        this.setName("Student");  
    }  
}
```

**ERROR: Implicit super constructor Person() is undefined.  
Must explicitly invoke another constructor**



```
public class Person {  
    private String name;  
    public Person( String n ) {  
        super();  
        this.name = n;  
    }  
    public void setName( String n) {  
        this.name = n;  
    }  
}
```

```
Student s = new Student();
```

super()



The diagram consists of two yellow lines originating from a yellow box labeled 'super()'. One line points to the 'super()' call inside the 'Person' class constructor. The other line points to the 'Student' class constructor in the code block below, illustrating that the 'Student' constructor inherits the 'super()' call from its parent class 'Person'.

```
public class Student extends Person {  
    public Student () {  
        this.setName("Student");  
    }  
}
```