**Question 5 Answer:** The state printed for the Undergraduate object **u1** comes from the **toString()** method inherited from the parent class **Student**.

**Question 8 Answer:** Not all of the state printed properly becausee the **Undergraduate** class did not have attributes such as **minor**, **major**, and **credits** at that point.

**Question 10 Answer:** After refining the **toString()** method in the **Undergraduate** class, all the state printed properly. The difference between objects **s1** and **s2** compared to **u1**, **u2**, and **u3** is that the latter include additional information related to major, minor, and credits.

**Bonus Question Answer:** The code still works because the **toString()** methods are inherited from the parent class. If not present in the subclass, the parent class's method is invoked automatically. This is possible due to polymorphism.

**Question 12 Answer:** No, the code for the **Student** objects **s1** and **s2** did not need to be changed in the **StudentApp** class. The new constructor is overloaded, so the existing code still works with the default constructor.

**Question 13 Answer:** When you compile the **Undergraduate** class, the superclass constructor **Student()** with no arguments is called by the **Undergraduate** constructor.

**Question 14 Answer:** Commenting off the no-argument constructor in the **Student** class causes a compilation error in the **Undergraduate** class because the default constructor is implicitly invoked, and it's no longer available. To fix this, either provide a parameterized constructor in the **Student** class or use the existing one.

**Question 20 Answer:** After changing the status of the **Undergraduate** and **Postgraduate** objects and calculating their fees, you observe that fees are calculated differently based on the overridden **calculateFees()** methods in the respective subclasses.

**Question 21 Answer:** When you invoke the **calculateFees()** method on the **Student** objects **s1** and **s2**, it calculates fees based on the default implementation in the parent class, as the method is not overridden in the subclasses.