AP® Computer Science A
2004 Sample Student Responses

A2 A1

(a) Given the class hierarchy shown above, write a complete class declaration for the class `Cat`, including implementations of its constructor and method(s). The `Cat` method `speak` returns `"meow"` when it is invoked.

```
public class Cat extends Pet
{
    public Cat (String name)
    {
        super(name);
    }
    public String speak()
    {
        return "meow";
    }
}
```

Part (b) begins on page 10.

A2

(b) Assume that class `Dog` has been declared as shown at the beginning of the question. If the `String` *dog-sound* is returned by the `Dog` method `speak`, then the `LoudDog` method `speak` returns a `String` containing *dog-sound* repeated two times.

Given the class hierarchy shown previously, write a complete class declaration for the class `LoudDog`, including implementations of its constructor and method(s).

```
public class LoudDog extends Dog
{
    publiz LoudDog (String name)
    {
        super(name);
    }

    publiz String speak()
    {
        return super.speak() + "  " + super.speak();
    }
}
```

A3

(c) Consider the following partial declaration of class Kennel.

```
public class Kennel
{
  [redacted]                    // all elements are references
                                // to Pet objects

  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  [redacted]
  {  /* to be implemented in this part */  }

  // ... constructor and other methods not shown
}
```

Write the Kennel method allSpeak. For each Pet in the kennel, allSpeak prints a line with the name of the Pet followed by the result of a call to its speak method.

In writing allSpeak, you may use any of the methods defined for any of the classes specified for this problem. Assume that these methods work as specified, regardless of what you wrote in parts (a) and (b). Solutions that reimplement functionality provided by these methods, rather than invoking these methods, will not receive full credit.

Complete method allSpeak below.

```
// postcondition: for each Pet in the kennel, its name followed
//                by the result of a call to its speak method
//                has been printed, one line per Pet
public void allSpeak()
{
    for (int i = 0; i < petList.size(); i ++)
    {
        Pet hold = (Pet)petList.get(i);
        System.out.println(hold.getName() + " " + hold.speak());
    }
}
```

GO ON TO THE NEXT PAGE.

A2 B1

(a) Given the class hierarchy shown above, write a complete class declaration for the class `Cat`, including implementations of its constructor and method(s). The `Cat` method `speak` returns `"meow"` when it is invoked.

```
public class Cat extends Pet
{
    public Cat (String name)
        myName = name;
    public String speak()
        return "meow";
}
```

Part (b) begins on page 10.

(b) Assume that class `Dog` has been declared as shown at the beginning of the question. If the `String` *dog-sound* is returned by the `Dog` method `speak`, then the `LoudDog` method `speak` returns a `String` containing *dog-sound* repeated two times.

Given the class hierarchy shown previously, write a complete class declaration for the class `LoudDog`, including implementations of its constructor and method(s).

```
public class LoudDog extends Dog
{
    public LoudDog (String name)
        myName = name;
    public String speak()
        return dog-sound + dog-sound;
}
```

(c) Consider the following partial declaration of class `Kennel`.

```
public class Kennel
{
  [redacted]                    // all elements are references
                                // to Pet objects

  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  [redacted]
  {  /* to be implemented in this part */  }

  // ... constructor and other methods not shown
}
```

Write the `Kennel` method `allSpeak`. For each `Pet` in the kennel, `allSpeak` prints a line with the name of the `Pet` followed by the result of a call to its `speak` method.

In writing `allSpeak`, you may use any of the methods defined for any of the classes specified for this problem. Assume that these methods work as specified, regardless of what you wrote in parts (a) and (b). Solutions that reimplement functionality provided by these methods, rather than invoking these methods, will not receive full credit.

Complete method `allSpeak` below.

```
// postcondition: for each Pet in the kennel, its name followed
//                by the result of a call to its speak method
//                has been printed, one line per Pet
public void allSpeak()
{ for (int i=0; i < petList.size(); i++)
    System.out.println(petList.get(i).getName() + "
                       petList.get(i).speak());
```

GO ON TO THE NEXT PAGE.

A2 C1

(a) Given the class hierarchy shown above, write a complete class declaration for the class `Cat`, including implementations of its constructor and method(s). The `Cat` method `speak` returns `"meow"` when it is invoked.

```
public class Cat extends Pet
{
    private String cName;
    public Cat (String name)
    {
        cName = name;
    }

    public String speak()
    {
        return "meow";
    }
}
```

Part (b) begins on page 10.

(b) Assume that class `Dog` has been declared as shown at the beginning of the question. If the `String` *dog-sound* is returned by the `Dog` method `speak`, then the `LoudDog` method `speak` returns a `String` containing *dog-sound* repeated two times.

Given the class hierarchy shown previously, write a complete class declaration for the class `LoudDog`, including implementations of its constructor and method(s).

```
public class LoudDog extends Dog
{
    private String dName;
    public Dog (String name)
    {
        dName = name;

    }
    public String speak()
    {
        String dogsound = "bark bark";
        return dogsound;
    }
```

GO ON TO THE NEXT PAGE.

C3

(c) Consider the following partial declaration of class `Kennel`.

```
public class Kennel
{
  [redacted]                    // all elements are references
                                // to Pet objects

  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  [redacted]
  {  /* to be implemented in this part */  }

  // ... constructor and other methods not shown
}
```

Write the `Kennel` method `allSpeak`. For each `Pet` in the kennel, `allSpeak` prints a line with the name of the `Pet` followed by the result of a call to its `speak` method.

In writing `allSpeak`, you may use any of the methods defined for any of the classes specified for this problem. Assume that these methods work as specified, regardless of what you wrote in parts (a) and (b). Solutions that reimplement functionality provided by these methods, rather than invoking these methods, will not receive full credit.

Complete method `allSpeak` below.

```
  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  public void allSpeak()
  {
      for (int x = 0; x < petList.size(); x++)
      {
          System.out.println(petList.get(x).getName() + "  " +
                             petList.get(x).speak());
      }
  }
```

GO ON TO THE NEXT PAGE.

A2 D1

(a) Given the class hierarchy shown above, write a complete class declaration for the class `Cat`, including implementations of its constructor and method(s). The `Cat` method `speak` returns `"meow"` when it is invoked.

```
public class Cat extends Pet
{
  public Cat (string Name)
  {
    super. myName;
  }
  public string speak ()
  {
    return "meow";
  }
}
```

Part (b) begins on page 10.

(b) Assume that class `Dog` has been declared as shown at the beginning of the question. If the `String` *dog-sound* is returned by the `Dog` method `speak`, then the `LoudDog` method `speak` returns a `String` containing *dog-sound* repeated two times.

Given the class hierarchy shown previously, write a complete class declaration for the class `LoudDog`, including implementations of its constructor and method(s).

```
public class LoudDog extends Dog
{
  public LoudDog (string name)
  { super. myName;
  }
  public string Speak()
  { super. Speak;
    return Speak + Speak;
  }
}
```

(c) Consider the following partial declaration of class `Kennel`.

```
public class Kennel
{
  [redacted]                    // all elements are references
                                // to Pet objects

  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  [redacted]
  {  /* to be implemented in this part */  }

  // ... constructor and other methods not shown
}
```

Write the `Kennel` method `allSpeak`. For each `Pet` in the kennel, `allSpeak` prints a line with the name of the `Pet` followed by the result of a call to its `speak` method.

In writing `allSpeak`, you may use any of the methods defined for any of the classes specified for this problem. Assume that these methods work as specified, regardless of what you wrote in parts (a) and (b). Solutions that reimplement functionality provided by these methods, rather than invoking these methods, will not receive full credit.

Complete method `allSpeak` below.

```
  // postcondition: for each Pet in the kennel, its name followed
  //                by the result of a call to its speak method
  //                has been printed, one line per Pet
  public void allSpeak()
  {
    super.myName;
    super.speak;
    return myName + " " + speak + " " + myName + " " + speak + " " + myName
           + " " + speak + " " + myName + " " + speak;
  }
```

GO ON TO THE NEXT PAGE.