Conrad Horton
CSC203 Jun 2016
Classes and Objects Lab
20160706

Part 1

1. Why would a class provide overloaded constructors?

For circumstances where you need more or less data when instantiating an object.

2. What are some advantages of creating packages?

Packages behave like a namespace. They provide encapsulated functionality, and make it so names do not conflict with other class names or methods. It also bundles up code, so if it is not needed it will not be included.

3. What is the purpose of a constructor?

To instantiate an instance of a class.

4. What is the purpose of a set method?

Provides encapsulated write functionality to non-public data members.

5. What is the purpose of a get method?

Provides read functionality to non-public data members.

6. What is an abstract data type?

Abstract classes cannot have an instance created of it. It must be defined in a subclass.

Part 2

```java
// Fig. 8.5: Time2.java
// Time2 class declaration with overloaded constructors.

public class Time2
{
   private int hour;   // 0 - 23
   private int minute; // 0 - 59
   private int second; // 0 - 59

   // Time2 no-argument constructor: initializes each instance variable
   // to zero; ensures that Time2 objects start in a consistent state
   public Time2()
   {
      this( 0, 0, 0 ); // invoke Time2 constructor with three arguments
   } // end Time2 no-argument constructor

   // Time2 constructor: hour supplied, minute and second defaulted to 0
   public Time2( int h )
   {
      this( h, 0, 0 ); // invoke Time2 constructor with three arguments
   } // end Time2 one-argument constructor

   // Time2 constructor: hour and minute supplied, second defaulted to 0
   public Time2( int h, int m )
   {
      this( h, m, 0 ); // invoke Time2 constructor with three arguments
   } // end Time2 two-argument constructor

   // Time2 constructor: hour, minute and second supplied
   public Time2( int h, int m, int s )
   {
      setTime( h, m, s ); // invoke setTime to validate time
   } // end Time2 three-argument constructor

   // Time2 constructor: another Time2 object supplied
   public Time2( Time2 time )
   {
      // invoke Time2 three-argument constructor
      this( time.getHour(), time.getMinute(), time.getSecond() );
   } // end Time2 constructor with a Time2 object argument

```

```
42      // Set Methods
43      // set a new time value using universal time; ensure that
44      // the data remains consistent by setting invalid values to zero
45      public void setTime( int h, int m, int s )
46      {
47         setHour( h );   // set the hour
48         setMinute( m ); // set the minute
49         setSecond( s ); // set the second
50      } // end method setTime
51
52      // validate and set hour
53      public void setHour( int h )
54      {
55         hour = ( ( h >= 0 && h < 24 ) ? h : 0 );
56      } // end method setHour
57
58      // validate and set minute
59      public void setMinute( int m )
60      {
61         minute = ( ( m >= 0 && m < 60 ) ? m : 0 );
62      } // end method setMinute
63
64      // validate and set second
65      public void setSecond( int s )
66      {
67         second = ( ( s >= 0 && s < 60 ) ? s : 0 );
68      } // end method setSecond
69
70      // Get Methods
71      // get hour value
72      public int getHour()
73      {
74         return hour;
75      } // end method getHour
76
77      // get minute value
78      public int getMinute()
79      {
80         return minute;
81      } // end method getMinute
82
83      // get second value
84      public int getSecond()
85      {
86         return second;
87      } // end method getSecond
88
89      // convert to String in universal-time format (HH:MM:SS)
90      public String toUniversalString()
91      {
92         return String.format(
93            "%02d:%02d:%02d", getHour(), getMinute(), getSecond() );
94      } // end method toUniversalString
95
96      // convert to String in standard-time format (H:MM:SS AM or PM)
97      public String toString()
98      {
99         return String.format( "%d:%02d:%02d %s",
100           ( (getHour() == 0 || getHour() == 12) ? 12 : getHour() % 12 ),
101           getMinute(), getSecond(), ( getHour() < 12 ? "AM" : "PM" ) );
102     } // end method toString
103  } // end class Time2
```

1) What is output by the following code segment?

```
1  Time3 t1 = new Time3( 5 );
2  System.out.printf( "The time is %s\n", t1 );
```

Time3 not defined.

2) What is output by the following code segment?

```
1  Time3 t1 = new Time3( 13, 59, 60 );
2  System.out.printf( "The time is %s\n", t1 );
```

Time3 not defined.

3) What is output by the following code segment?

```
1  Time3 t1 = new Time3( 0, 30, 0 );
2  Time3 t2 = new Time3( t1 );
3  System.out.printf( "The time is %s\n", t2.toUniversalString() );
```

Time3 not defined.

```java
public class Person
{
   private String firstName;
   private String lastName;
   private String gender;
   private int age;

   public Person( String firstName, String lastName )
   {
      setName( firstName, lastName );
      setGender( "n/a" );
      setAge( -1 );
   } // end Person constructor

   public Person( String firstName, String lastName, String gender, int age )
   {
      setName( firstName, lastName );
      setGender( gender );
      setAge( age );
   } // end Person constructor

   public void setName( String firstName, String lastName )
   {
      this.firstName = firstName;
      this.lastName = lastName;
   } // end method setName

   public void setGender( String gender )
   {
      this.gender = gender;
   } // end method setGender

   public void setAge( int age )
   {
      this.age = age;
   } // end method setAge

   public String getName()
   {
      return String.format( "%s %s", firstName, lastName );
   } // end method getName

   public String getGender()
   {
      return gender;
   } // end method getGender

   public int getAge()
   {
      return age;
   } // end method getAge

   public String toString()
   {
      if ( gender == "n/a" && age == -1 )
         return getName();

      return String.format( "%s is a %d year old %s", getName(), getAge(),
         getGender() );
   } // end method toString
}  // end class Person
```

1.

```
Person person = new Person( "Rus", "Tic", "male", 21 );
System.out.println( person );
```

Rus Tic is a 21 year old male.

2.

```
Person person = new Person( "Anna Lee", "Tic" );
System.out.println( person );
```

Anna Lee Tic

3.

```
Person person = new Person( "Anna Lee", "Tic", "n/a", -1 );
System.out.println( person );
```

Anna Lee Tic