```java
/* Class:   ExtClock.java
 * Author:  Instructor
 * Purpose: Solution to Assignment 01 programming lab.
 *          Implements an ExtClock by adding an integer time zone to Clock
 *          (follows standard Java naming and indentng).
 */

public class ExtClock extends Clock
{
   private int zone; // in range [-12, 12]

   //Constructor with parameters, to set the time
   //The time is set according to the parameters
   //Precondition:  valid values for hours, minutes, seconds and zone
   //Postcondition: hr = hours; min = minutes; sec = seconds; zone = tz
   public ExtClock(int hours, int minutes, int seconds, int tz)
   {
      setTime(hours, minutes, seconds, tz);
   }

   //Default constructor, time is set to midnight GMT
   //Postcondition: hr = 0; min = 0; sec = 0; zone = 0
   public ExtClock()
   {
      setTime(0, 0, 0, 0);
   }

   //Method to set the time
   //The time is set according to the parameters
   //Precondition:  valid values for hours, minutes, seconds and zone
   //Postcondition: hr = hours; min = minutes; sec = seconds; zone = tz
   public void setTime(int hours, int minutes, int seconds, int tz)
   {
      setTime(hours, minutes, seconds);
      setZone(tz);
   }

   //Method to set the zone
   //The Zone is set according to the parameter
   //Precondition:  integer in range [-12, 12]
   //Postcondition: zone = tz
   public void setZone(int tz)
   {
      if (tz >= -12 && tz <= +12) zone = tz;
      else zone = 0;
   }

   //Method to return the zone
   //Postcondition: the value of zone is returned
   public int getZone()
   {
      return zone;
   }

   //Method to print the time
   //Postcondition: Time is printed in the form hh:mm:ss UTC+z
   public void printTime()
   {
      super.printTime();
      System.out.printf(" UTC%+d", zone);
   }

   //Method to compare this time to another time
   //Precondition:  otherClock references a valid ExtClock object instance
   //Postcondition: returns true if time is equal to otherClock
   //               returns false otherwise
   public boolean equals(ExtClock otherClock)
   {
      return(super.equals(otherClock) && zone == otherClock.zone);
   }
```

```
71
72      //Method to copy another time into this one
73      //Precondition:  otherClock references a valid ExtClock object instance
74      //Postcondition: The data members of otherClock are copied
75      //               into the correspoding data members of this ExtClock
76      public void makeCopy(ExtClock otherClock)
77      {
78         super.makeCopy(otherClock);
79         zone = otherClock.zone;
80      }
81
82      //Method to return a copy of this time
83      //Postcondition: The data members of this ExtClock are copied
84      //               into the correspoding data members of a new ExtClock
85      public Clock getCopy()
86      {
87         ExtClock temp = new ExtClock();
88         temp.makeCopy(this);
89         return temp;
90      }
91
92      //Method to change the zone, adjusting the time accordingly
93      //Precondition:  integer in range [-12, 12]
94      //Postcondition: The time in this ExtClock is adjusted to the corresponding
95      //               time in the new zone and this zone is set to the new zone
96      public void setUTC(int tz)
97      {
98         if (tz < -12 || tz > +12) tz = 0;
99         int newHours = getHours()-(zone - tz);
100        if (newHours < 0) newHours += 24;
101        setTime(newHours%24, getMinutes(), getSeconds(), tz);
102     }
103
104     //Method to override toString to return a standard UTC time with its zone
105     //Postcondition: Time is returned in the form hh:mm:ss UTC+z
106     public String toString()
107     {
108        String str = super.toString();
109        str = str + String.format(" UTC%+d", zone);
110        return str;
111     }
112  }
```