File - C:\Users\ctqdt\IdeaProjects\CourseScheduler\src\com\company\data\person\PersonAddress.java

```java
/**
 * @author Caroline Ta
 * @since 05.19.2020
 */
package com.company.data.person;

import org.jetbrains.annotations.NotNull;

/**
 * The type Person address.
 */
public class PersonAddress implements Comparable<PersonAddress>{
    /**
     * The street.
     */
    private String street;
    /**
     * The city.
     */
    private String city;
    /**
     * The state.
     */
    private String state;
    /**
     * The zip code.
     */
    private String zip;

    // ------------------------------------------------------------------------
    // CONSTRUCTORS
    // ------------------------------------------------------------------------

    /**
     * Instantiates a new Person address.
     */
    public PersonAddress() {

    }

    /**
     * Instantiates a new Person address.
     *
     * @param street the street
     * @param city   the city
     * @param state  the state
     * @param zip    the zip
     * @throws Exception the exception
     */
    public PersonAddress(String street, String city, String state, String zip) throws Exception {
        setStreet(street);
        setCity(city);
        setState(state);
        setZip(zip);
    }

    // ------------------------------------------------------------------------
    // ACCESSORS - GETTER METHODS
    // ------------------------------------------------------------------------

    /**
     * Gets street.
     *
     * @return the street
     */
    public String getStreet() {
        return street;
    }

    /**
```

```
71       * Gets city.
72       *
73       * @return the city
74       */
75      public String getCity() {
76          return city;
77      }
78
79      /**
80       * Gets state.
81       *
82       * @return the state
83       */
84      public String getState() {
85          return state;
86      }
87
88      /**
89       * Gets zip.
90       *
91       * @return the zip
92       */
93      public String getZip() {
94          return zip;
95      }
96
97      // ------------------------------------------------------------------------
98      // MUTATORS - SETTER METHODS
99      // ------------------------------------------------------------------------
100
101     /**
102      * Sets street.
103      *
104      * @param street the street
105      * @throws Exception the exception
106      */
107     public void setStreet(String street) throws Exception {
108         checkStringValue(street, "street");
109         this.street = street;
110     }
111
112     /**
113      * Sets city.
114      *
115      * @param city the city
116      * @throws Exception the exception
117      */
118     public void setCity(String city) throws Exception {
119         checkStringValue(city, "city");
120         this.city = city;
121     }
122
123     /**
124      * Sets state.
125      *
126      * @param state the state
127      * @throws Exception the exception
128      */
129     public void setState(String state) throws Exception {
130         checkStringValue(state, "state");
131         this.state = state;
132     }
133
134     /**
135      * Sets zip.
136      *
137      * @param zip the zip
138      * @throws Exception the exception
139      */
140     public void setZip(String zip) throws Exception {
```

```
141         checkStringValue(zip, "zip");
142         this.zip = zip;
143     }
144 
145     // --------------------------------------------------------------------------
146     // FUNCTIONALITY METHODS
147     // --------------------------------------------------------------------------
148 
149     /**
150      * Check string value.
151      *
152      * @param valueString   the value string
153      * @param attributeName the attribute name
154      * @throws Exception the exception
155      */
156     private void checkStringValue(String valueString, String attributeName) throws Exception {
157         if (valueString == null || valueString.trim().equals("")) {
158             throw new Exception(attributeName + " must be non-null and non-empty");
159         }
160     }
161 
162     // --------------------------------------------------------------------------
163     // OVERRIDDEN METHODS
164     // --------------------------------------------------------------------------
165 
166     /**
167      * Override toString method.
168      *
169      * @return a string description for the PersonAddress class.
170      */
171     @Override
172     public String toString()
173     {
174         return street + " St, " + city + ", " + state + ", " + zip;
175     }
176 
177     /**
178      * Override compareTo method.
179      *
180      * @param otherPerson other person address
181      * @return comparision result (i.e -1, 0) between PersonAddresses
182      */
183     @Override
184     public int compareTo(@NotNull PersonAddress otherPerson) {
185         if (this.street.equalsIgnoreCase(otherPerson.street)
186                 && this.city.equalsIgnoreCase(otherPerson.city)
187                 && this.state.equalsIgnoreCase(otherPerson.state)
188                 && this.zip == otherPerson.zip) {
189             return 0;
190         }
191 
192         else
193         {
194             return -1;
195         }
196     }
197 
198     /**
199      * Override equals method.
200      *
201      * @param obj object
202      * @return the result of thisPersonAddress equals otherPersonAddress as a boolean
203      */
204     @Override
205     public boolean equals(Object obj) {
206         if (obj == null || !(obj instanceof PersonAddress)) {
207             return false;
208         }
209 
210         return compareTo((PersonAddress) obj) == 0;
```

```
211         }
212 }
213
```