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

import java.io.Serializable;

/**
 * The type Person.
 */
public abstract class Person implements Comparable<Person>, Cloneable, Serializable {

    /**
     * The person name.
     */
    private PersonName name;
    /**
     * The person email.
     */
    private String email;
    /**
     * The person telephone number.
     */
    private String tel;
    /**
     * The person address information.
     */
    private PersonAddress addressInfo;

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

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

    }

    /**
     * Instantiates a new Person.
     *
     * @param nameO    the name object
     * @param email    the email
     * @param tel      the tel
     * @param addressO the address object
     */
    public Person(PersonName nameO, String email, String tel, PersonAddress addressO) {
        setName(nameO);
        setEmail(email);
        setTel(tel);
        setAddressInfo(addressO);
    }

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

    /**
     * Gets name.
     *
     * @return the name
     */
    public String getName() {
        return name.toString();
    }

    /**
```

```java
     * Gets email.
     *
     * @return the email
     */
    public String getEmail() {
        return email;
    }

    /**
     * Gets telephone number.
     *
     * @return the telephone number
     */
    public String getTel() {
        return tel;
    }

    /**
     * Gets address info.
     *
     * @return the address info
     */
    public String getAddressInfo() {
        return addressInfo.toString();
    }

    // -------------------------------------------------------------------------
    // MUTATORS - SETTER METHODS
    // -------------------------------------------------------------------------

    /**
     * Sets name.
     *
     * @param name the name
     */
    public void setName(PersonName name) {
        this.name = name;
    }

    /**
     * Sets email.
     *
     * @param email the email
     */
    public void setEmail(String email) {
        this.email = email;
    }

    /**
     * Sets tel.
     *
     * @param tel the tel
     */
    public void setTel(String tel) {
        this.tel = tel;
    }

    /**
     * Sets address info.
     *
     * @param addressInfo the address info
     */
    public void setAddressInfo(PersonAddress addressInfo) {
        this.addressInfo = addressInfo;
    }

    // -------------------------------------------------------------------------
    // FUNCTIONALITY METHODS
    // -------------------------------------------------------------------------

    /**
     * Check string value.
     *
     * @param valueString   the value string
     * @param attributeName the attribute name
     * @throws Exception the exception
     */
    protected void checkStringValue(String valueString, String attributeName) throws Exception {
        if (valueString == null || valueString.trim().equals("")) {
            throw new Exception(attributeName + " must be non-null and non-empty");
        }
    }

    // -------------------------------------------------------------------------
    // OVERRIDDEN METHODS
    // -------------------------------------------------------------------------

    /**
     * Override toString method.
     *
     * @return a string description of the Person class.
     */
    @Override
    public String toString() {
        StringBuilder personOutput = new StringBuilder();

        personOutput.append("\nName:           ");
        personOutput.append(name);

        personOutput.append("\nEmail:          ");
        personOutput.append(email);

        personOutput.append("\nTelephone:      ");
        personOutput.append(tel);

        personOutput.append("\nAddress:        ");
        personOutput.append(addressInfo);

        return personOutput.toString();
    }

    /**
     * Override compareTo method.
     *
     * @param o person object
     * @return comparision result (i.e. -1, 0, 1) between Persons
     */
    @Override
    public int compareTo(Person o) {
        return this.tel.compareTo(o.tel);
    }

    /**
     * Override equals method.
     *
     * @param obj the object
     * @return the result of thisCourse equals otherCourse as a boolean
     */
    @Override
    public boolean equals(Object obj) {
        if (obj == null || !(obj instanceof Person)) {
            return false;
        }

        return compareTo((Person) obj) == 0;
    }

    /**
     * Override clone method.
     *
     * @return super clone
     * @throws CloneNotSupportedException Clone Not Supported Exception
     */
    @Override
    public Object clone() throws CloneNotSupportedException {
        return super.clone();
    }

    /**
     * Override hashCode method.
     *
     * @return hashCode of telephone number.
     */
    @Override
    public int hashCode()
    {
        return tel.hashCode();
    }

}

```