** COMSATS University Islamabad, Lahore Campus**

***Lab 8***

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Title: | Object Oriented Software Engineering | | | | Course Code: | CSC 304 | Credit Hours: | | 3(2,1) |
| Course Instructor IndhsdjsIInstructorInstructor/s: | Yella Mehroze | | | | Theory/Lab Name: | Lab | | | |
| Semester: | 4th | Batch: | Fa19-BSE | Section: | A , B | Semester | | Spring 2021 | |

Student Name – AREEB AHMED

Registration ID- FA19-BSE-022

Section -B

**Outline**

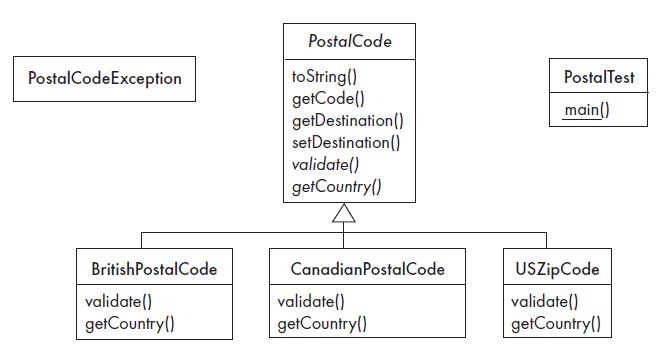
**Learning Outcomes for This Lab**

* + **Separation of user interface from functional part of the system**
  + **Illustration and revision of following Java concepts:**
    - **Inheritance**
    - **Polymorphism**
    - **String Manipulation**
    - **Access Control**

**[Write your solutions in this document]**

# **Task 1**

Implement the following class diagram. Classes for the manipulation of postal code and public methods are being displayed in the diagram. For the test class, you should write the code in a manner that allows the user to enter the postal code and check the facilities being provided by the postal code hierarchy as illustrated in the diagram. [For reference or further details, see the page number 55 of your course text book].

******

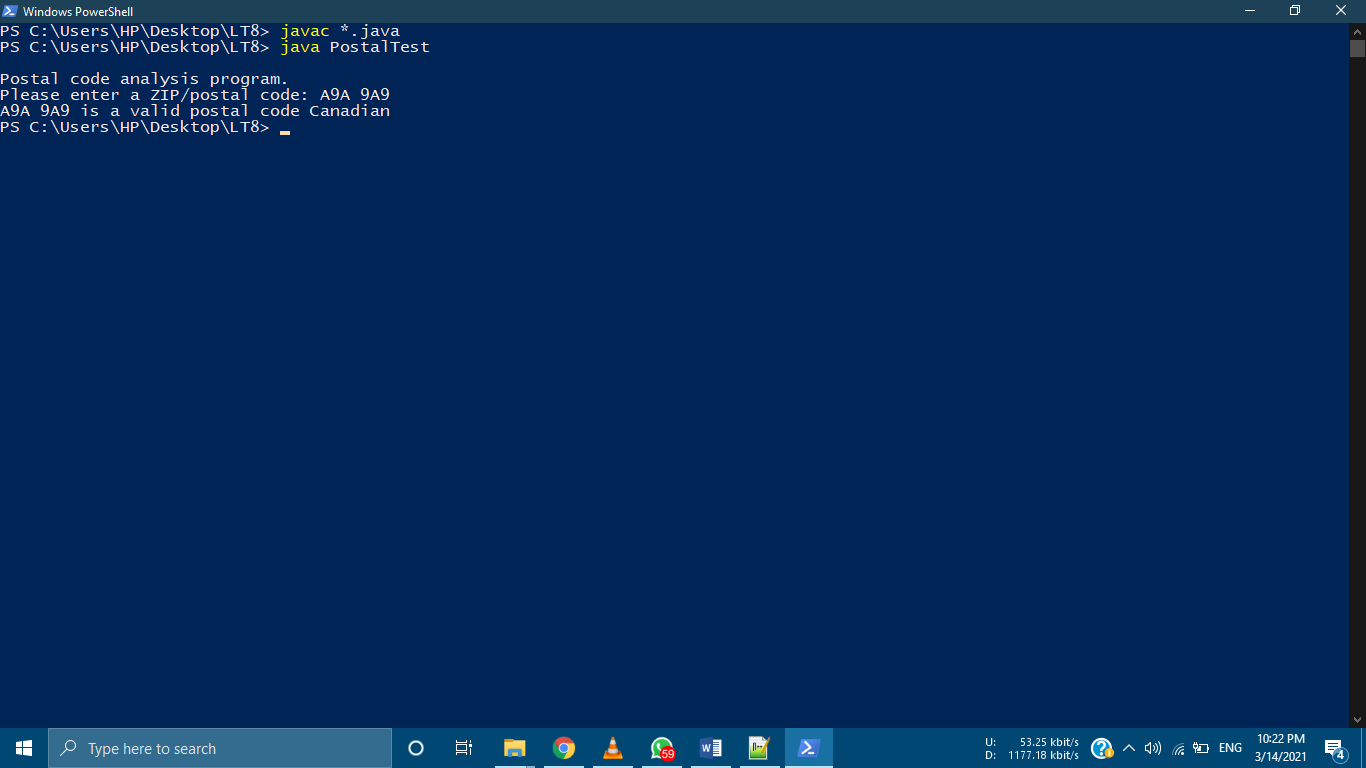
# **Task 2**

After the implementation of hierarchy illustrated in above class diagram, execute the following questions designed for this hierarchy. [Hint – Have a look at page number 52 and 53 of your text book for the revision of concepts].

***QUESTION: 1***

Run the postal code program. Then carefully read through the code for all six classes. Use the Java documentation to look up any methods or classes you do not understand.

***Answer***



***QUESTION: 2***

The way the program is written, letters in Canadian postal codes are only accepted if they are upper case. On the other hand, letters in British postal codes are accepted whether they are upper case or not. This is inconsistent. Modify the program so that user input of upper or lower case is accepted, and the input is converted to upper case immediately.

***Answer***

protected void validate() throws PostalCodeException

{

String p\_code = getCode();

if ((p\_code.length()== 0)

|| (!Character.isLetter(p\_code.charAt(0)))

|| (!Character.isDigit(p\_code.charAt(1)))

|| (!Character.isLetter(p\_code.charAt(2)))

|| (!Character.isWhitespace(p\_code.charAt(3)))

|| (!Character.isDigit(p\_code.charAt(4)))

|| (!Character.isLetter(p\_code.charAt(5)))

|| (!Character.isDigit(p\_code.charAt(6)))

|| (p\_code.length() > 7))

{

throwException("Sequence of characters not like B8B 8B8");

}

}

***QUESTION: 3***

Describe how you would design the following modifications to the postal code program. Think carefully about whether there should be one method, or several different polymorphic methods. In the latter case, think about whether there should be an abstract method in the superclass and concrete methods in the subclasses, or else a concrete method in the superclass and one or more overriding methods in the subclasses.

(a) There should be an operation length that returns the number of characters in a postal code.

(b) There should be a file that contains postal codes, one per line. There should then be an operation called isOnRecord that returns true if a postal code is in this file. Do not worry for now about the efficiency of this operation in the case of very large files, although you should be aware that this would be a concern in a production-quality system. Hint: investigate class FileInputStream.

(c) For each country, there should be a file that contains, on each line, a postal code prefix followed by the name of a destination of such postal codes. For example, class BritishPostalCode might use the file BritishPostalDestinations.txt, and on one of its lines it might contain ‘SW Southwest-London’. The parts of the program that set the destination should read these files.

***Answer***

***QUESTION: 4***

Implement the designs you prepared in the above exercise.

***Answer***

***QUESTION: 5***

Add a new subclass representing postal codes for our beloved country Pakistan. Assume that the format is always one or two letters, followed by a space, followed by two numbers. You will have to modify the PostalTest class to accommodate your new subclass, although you must not modify the PostalCode class.

***Answer***

class PakistanCode extends PostalCode

{

public PakistanCode(String code) throws PostalCodeException

{

super(code);

}

public String getCountry()

{

return "Pakistan";

}

String p\_code = getCode();

if(p\_code.length() < 5)

throwException("Postal code too short");

int var = 0;

if(!Character.isLetter(p\_code.charAt(0)))

throwException("Expecting letter at position 1");

var++;

if(Character.isLetter(p\_code.charAt(1)))

var++;

if(!Character.isWhitespace(p\_code.charAt(var)))

throwException("Expecting space at position "+(var+1));

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if(p\_code.length() > var)

throwException("Unexpected character at end of code");

setDestination("outside Pakistan");

String[] pakistanCodes = {"LH", "KR", "GW", "RP", "QT", "PW"};

for(int i=0; i<pakistanCodes.length; i++)

{

if(p\_code.startsWith(pakistanCodes[i]))

{

setDestination("within Pakistan.");

return;

}

}

}

# **Task 3**

Paste your code of all classes here. Highlight your class names, use proper indentation / naming conventions while writing your code. Upload your java files along with this task in Google Class Room. In every Java Class, write your name and registration ID.

**Answer:**

import java.io.\*;

abstract class PostalCode //Areeb Ahmed(FA19-BSE-022)

{

private String code;

private boolean hasBeenValidated = false;

private boolean isValid = true;

private String destination;

public PostalCode(String code) throws PostalCodeException

{

this.code = code;

this.isValid = isValid();

}

public String toString()

{

return code + " is a"

+ ((isValid) ? (" valid postal code " + getCountry()) : "n invalid");

}

public String getCode()

{

return code;

}

public String getDestination()

{

return destination;

}

public void setDestination(String destination)

{

this.destination = destination;

}

public boolean isValid() throws PostalCodeException

{

if(!hasBeenValidated)

{

hasBeenValidated = true;

validate();

}

return isValid;

}

void throwException(String message)

throws PostalCodeException

{

isValid = false;

PostalCodeException e =

new PostalCodeException("Exception: " + message);

throw e;

}

protected abstract void validate() throws PostalCodeException;

public abstract String getCountry();

}

class BritishPostalCode extends PostalCode //Areeb Ahmed(FA19-BSE-022)

{

public BritishPostalCode(String code) throws PostalCodeException

{

super(code);

}

public String getCountry()

{

return "Britain";

}

protected void validate() throws PostalCodeException

{

String p\_code = getCode();

if(p\_code.length() < 6)

throwException("Postal code too short");

int var = 0;

if(!Character.isLetter(p\_code.charAt(0)))

throwException("Expecting letter at position 1");

var++;

if(Character.isLetter(p\_code.charAt(1)))

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if((Character.isDigit(p\_code.charAt(var))

|| (p\_code.charAt(var) == 'A' && p\_code.charAt(var-1) == '1')))

var++;

if(!Character.isWhitespace(p\_code.charAt(var)))

throwException("Expecting space at position "+(var+1));

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if(p\_code.length() > var

&& Character.isDigit(p\_code.charAt(var)))

var++;

for(int i=0; i<2; i++)

{

if(p\_code.length() <= var

||!Character.isLetter(p\_code.charAt(0)))

throwException("Expecting letter at position "+(var+1));

var++;

}

if(p\_code.length() > var)

throwException("Unexpected character at end of code");

setDestination("outside London.");

String[] londonCodes = {"NW", "NE", "SW", "SE", "EC", "WC"};

for(int i=0; i<londonCodes.length; i++)

{

if(p\_code.startsWith(londonCodes[i]))

{

setDestination("within London.");

return;

}

}

}

}

class CanadianPostalCode extends PostalCode //Areeb Ahmed(FA19-BSE-022)

{

public CanadianPostalCode(String code) throws PostalCodeException

{

super(code);

}

public String getCountry()

{

return "Canadian";

}

protected void validate() throws PostalCodeException

{

String p\_code = getCode();

if ((p\_code.length()== 0)

|| (!Character.isLetter(p\_code.charAt(0)))

|| (!Character.isDigit(p\_code.charAt(1)))

|| (!Character.isLetter(p\_code.charAt(2)))

|| (!Character.isWhitespace(p\_code.charAt(3)))

|| (!Character.isDigit(p\_code.charAt(4)))

|| (!Character.isLetter(p\_code.charAt(5)))

|| (!Character.isDigit(p\_code.charAt(6)))

|| (!Character.isUpperCase(p\_code.charAt(0)))

|| (!Character.isUpperCase(p\_code.charAt(2)))

|| (!Character.isUpperCase(p\_code.charAt(5)))

|| (p\_code.length() > 7))

{

throwException("Sequence of characters not like A9A 9A9");

}

}

}

class USZipCode extends PostalCode //Areeb Ahmed(FA19-BSE-022)

{

public USZipCode(String code) throws PostalCodeException

{

super(code);

}

public String getCountry()

{

return "American";

}

protected void validate() throws PostalCodeException

{

String p\_code = getCode();

switch(p\_code.length())

{

case 5:

case 10:

for (int i = 0; i < p\_code.length(); i++) //Check chars

{

if (((i == 5) && (p\_code.charAt(i) != '-'))

|| ((i != 5) && (!(Character.isDigit(p\_code.charAt(i))))))

{

throwException("Invalid character in code.");

break;

}

}

break;

default:

throwException("Invalid code length.");

}

setDestination(computeDestination());

}

private String computeDestination()

{

if (getCode().length() == 10)

return "in a major US city.";

else

return "in a small US city.";

}

}

class PostalCodeException extends Exception //Areeb Ahmed(FA19-BSE-022)

{

public PostalCodeException() {}

public PostalCodeException(String message)

{

super(message);

}

}

class PostalTest //Areeb Ahmed(FA19-BSE-022)

{

public static void main(String[] args)

{

PostalCode code = null;

String codeString = null;

System.out.println("\nPostal code analysis program.");

codeString = getInput(); //Get the code to analyse

try

{

code = new BritishPostalCode(codeString);

}

catch (PostalCodeException ex1)

{

try

{

code = new CanadianPostalCode(codeString);

}

catch (PostalCodeException ex2)

{

try

{

code = new USZipCode(codeString);

}

catch (PostalCodeException ex3) {}

}

}

if (code != null)

System.out.println(code);

else

System.out.println(codeString + " is an invalid ZIP or postal code.");

}

/\* class PakistanCode extends PostalCode

{

public PakistanCode(String code) throws PostalCodeException

{

super(code);

}

public String getCountry()

{

return "Pakistan";

}

String p\_code = getCode();

if(p\_code.length() < 5)

throwException("Postal code too short");

int var = 0;

if(!Character.isLetter(p\_code.charAt(0)))

throwException("Expecting letter at position 1");

var++;

if(Character.isLetter(p\_code.charAt(1)))

var++;

if(!Character.isWhitespace(p\_code.charAt(var)))

throwException("Expecting space at position "+(var+1));

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if(!Character.isDigit(p\_code.charAt(var)))

throwException("Expecting number at position "+(var+1));

var++;

if(p\_code.length() > var)

throwException("Unexpected character at end of code");

setDestination("outside Pakistan");

String[] pakistanCodes = {"LH", "KR", "GW", "RP", "QT", "PW"};

for(int i=0; i<pakistanCodes.length; i++)

{

if(p\_code.startsWith(pakistanCodes[i]))

{

setDestination("within Pakistan.");

return;

}

}

}

\*/

static private String getInput()

{

byte[] buffer = new byte[1024];

String myString = "";

System.out.print("Please enter a ZIP/postal code: ");

try

{

System.in.read(buffer);

myString = new String(buffer).trim();

myString.toUpperCase();

}

catch(Exception e){}

return myString;

}

}

***For any queries***

[***yellamehroze@cuilahore.edu.pk***](mailto:yellamehroze@cuilahore.edu.pk)

***Office: H-Block – Room 23 (H – 23)***