Lab Activity #1 – 22S_CST7284_300

Due date: *Wednesday May 18, 2022*

What to submit: One class called TelephoneOperator.java

The North American Numbering Plan (NANP) divides the territories of its member countries into Numbering Plan Areas (NPAs), each identified by a three-digit code commonly called area code. Twenty-four countries and territories share the North American Numbering Plan (NANP), with a single country code of 1 (CC). It is a closed telephone numbering plan in which all telephone numbers assigned to telephones consist of seven digits. In addition, each number has an associated area code consisting of three digits which is prefixed to the telephone number when dialing outside of the local numbering plan area. Multiple area codes may be overlaid to the same geographical area.

The formatting convention for phone numbers is (NPA) NXX-XXXX, where NPA is the numbering plan area code and NXX-XXXX is the subscriber number. The prefix NXX of the subscriber number is a code for the local central office, unique in the numbering plan area. The place holder N stands for the digits 2 to 9, as the subscriber number may not begin with the digits 0 and 1.

Write a class called **TelephoneOperator**. Within the main method, you need to verify the length of the phone number passed in. This class would need to have a set of overloaded methods called dialPhoneNumber which simply print out the appropriate message.

If the phone number is 3 digits, you call dialPhoneNumber which takes one int parameter called a special number (SpN). This method would print the following message “Dialing special number <SpN> …”. Special numbers (SpN) are such 311, 611, 911 … No need to verify anything except that the number consists of 3 digits. If not, the general error message needs to be displayed “Wrong phone number entered !”

If the phone number is 10 digits, you call dialPhoneNumber which takes two int parameters consisting of an NPA and a subscriber number (SN). This method would print the following message “Dialing local phone number <NPA> <subscriber-number> …” where NPA is 3 digits and SN has the format NXX-XXXX. You need to verify that N cannot be a 0 or a 1. If it is, this method would print instead the following message “Wrong local phone number entered !”. Pay attention to the hyphen (-) within the last seven digits.

If the phone number is 11 digits, you call dialPhoneNumber which takes three int parameters consisting of a CC, an NPA and a subscriber number (SN). This method would print the following message “Dialing NA long distance phone number <CC> <NPA> <SN> …” where CC is a 1, NPA is 3 digits and subscriber number has the format NXX-XXXX. Note the space between CC and NPA and the space between NPA and SN. You need to verify that CC is 1; if not, this method needs to display the following message “Wrong NA long distance phone number entered !”. NA stands for North America. You need also to verify that N cannot be a 0 or a 1. If it is, this method would print the same error message “Wrong NA long distance phone number entered !”. Pay attention to the hyphen (-) within the last seven digits.

If the number is 12 digits, you call dialPhoneNumber which takes four int parameters consisting of an international access code 011 (AC), a CC of three digits, municipality area code (MAC) consisting of 2

digits and an SN consisting of 4 digits. This method would print the following message “Dialing overseas long distance phone number <AC> <CC> <MAC>-<SN> …”. Note the spaces between each <>. You need to verify that AC equals 011, if not the following message needs to be displayed “Wrong overseas long distance phone number entered !”. No further verification is required on the rest of the codes.

You need to have a main method in this class which would create an instance of this class and uses it to invoke the methods required (refer to one of my class examples that displays this methodology). You cannot make any of the dialPhoneNumber methods static. This main method needs to have a loop that would keep prompting the user to enter a phone number and displays the appropriate message until a value of -1 is entered (check examples within slides).

Please make use of String methods like length, substring, charAt, indexOf... In order to get the hyphen (-) between the last seven digits when the number is 10 or 11, you would practice converting these last seven digits back into a String. One way is by using the static method String.valueOf(<int>) and then inserting the hyphen in its appropriate place. For the hyphen part, you can use StringBuffer method insert or simply sticking to methods within String class. There are many ways to do this and I will leave it up to you choose your desired methodology.

We will assume that no letters are entered; i.e. no need to verify for this. Also, any number that is entered that is not 3, 10, 11 or 12 in length, a general error message needs to be displayed which is “Wrong phone number entered !”

Sample output:

<Welcoming message to the Telephone Operator program>

Please enter a telephone number or -1 to exit: 911

Dialing special number 911 …

Please enter a telephone number or -1 to exit: 6135551212

Dialing local phone number 613 555-1212 …

Please enter a telephone number or -1 to exit: 6130551212

Wrong local phone number entered !

Please enter a telephone number or -1 to exit: 16130551212

Wrong NA long distance phone number entered !

Please enter a telephone number or -1 to exit: 16135551212

Dialing NA long distance phone number 1 613 555-1212 …

Please enter a telephone number or -1 to exit: 011963345678

Dialing overseas long distance phone number 011 963 34-5678 …".

Please enter a telephone number or -1 to exit: 012963345678

Wrong overseas long distance phone number entered !

Please enter a telephone number or -1 to exit: 0129

Wrong phone number entered !

Please enter a telephone number or -1 to exit: 012999

Wrong phone number entered !