Assignment 3 – Spring 2020

Morse code is a method used in telecommunication to encode text characters as standardized sequences of two different signal durations, called dots and dashes or dits and dahs. Morse code is named for Samuel Morse, an inventor of the telegraph. For further details, I refer to Wikipedia (https://en.wikipedia.org/wiki/Morse_code)

- ✓ ***Please do your own work, sharing and/or copying code and/or solution ideas with/from others will result in a grade of 0 and disciplinary actions for all involved parties. If you run into any problems and have done your best to solve them, please see me before/after class or e-mail me.***
- ✓ ***If your submission is late, you will incur a 20% deduction of your score for each late day.***

For this assignment, you will write a Java program to convert text (of selected characters) to Morse code using a sequence of dots and dashes. The program consists of the two classes shown in Figure 1. Your code MUST compile and run with the test code provided with the assignment ($TestMorseConverter.java$). DONOT MODIFY THE TEST CODE.

Notes:

1. Properly comment your code. Comments should precede variables, methods, and major steps in your code.
2. Follow the Java naming conventions and use the class and method names as shown in the class diagram.
3. File Input: *The java.util.Scanner class is a multi-purpose class. It can be used to read input from the command line or from a text file. For instance, the statement new* $Scanner(System.in)$ *creates an instance for reading input from the command line. The statement new* $Scanner(new\ File("file.dat"))$ *creates an instance for reading input from a text file called file.dat. After the instance is created, the behavior is the same. Use* $next(\ )$ *to read one String entry from the command line or the file; use* $nexInt(\ )$ *to read one Integer entry …*
4. Submit two $Java$ source files – do not submit the test class:
   1) $MorseCode.java$
   2) $MorseCodeConvert.java$
5. Running the sample JAR:
   1) Place the JAR and $TestFile.txt$ in the same directory. Create a new directory on your desktop and place the files there.
   2) Open a command prompt console and CD to the directory created in step 1
   3) Type the following command:

      java -jar Assignment3.jar

   4) Note that the path to $java$ must be already added to your system's path, otherwise you must provide the full path to the java executable

Classes minimum members:

- ✓ **Class $MorseCode$:**
  - ➢ ***Non-default constructor*:** accepts character values of ASCII codes between 32 and 90 and encoding values that are not NULL and of length at least one. If either criteria is NOT met, the constructor will throw an Exception with the message: "$The\ character \ll the\ invalid\ character \gg\ is\ not\ a\ supported\ Morse\ character$"
  - ➢ **Setters and getters**
- ✓ **Class $MorseCodeConvert$:**
  - ➢ $listCodes$: an $ArrayList$ to hold objects of type $MorseCode$. All valid encodings are held as in this list. Each object (character and encoding sequence) is represented as a single object before added to the list.

- **Non-default constructor:**
  - Accepts one parameter (the file to be read). The constructor attempts to open the file, if it fails, an exception is thrown with the message $Failed\ to\ open\ file: \ll\ file\ name\ \gg$
  - Read all lines from the file; add each valid line to the $ArrayList$. Each line in the input file consists of two columns separated by a tab (i.e. \t). Some lines in the file are corrupt, make sure you skip empty lines and lines that do not have exactly two entries. Your program should NOT halt on input errors; read the entire file.
  - If an exception from $MorseCode$ is encountered, print the exception message, do not add the character to the $ArrayList$, and continue to the next line in the input file.
  - For skipped lines, print the message $Invalid\ line: \ll\ the\ skipped\ line\ \gg$ and continue with the next line.
- ***printEncodingList*:** Prints the entire content of the $ArrayList$.
- ***encodeString***: *Accepts a string parameter and prints the corresponding Morse code for that string.*
- **encodeFile**:
  - Accepts a file name as input and prints the corresponding Morse code for the entire file's content.
  - Attempt to open the file, if it fails, an exception is thrown.
  - Read the contents of the file and print the corresponding Morse code.
  - If an invalid character is encountered, print '?' for its conversion.

Grading:

Item	Points
Comments	10
Class MorseCode	
Non-default constructor (char and encoding check)	15
Exception	5
Setters/Getters	10
Class MorseCodeConvert	
Non-default constructor: ✓ Exception ✓ File read ✓ Line skip ✓ Populate list	20
Exception	
printEncodingList()	10
encodeString()	10
encodeFile()	20
	100

Class Diagram

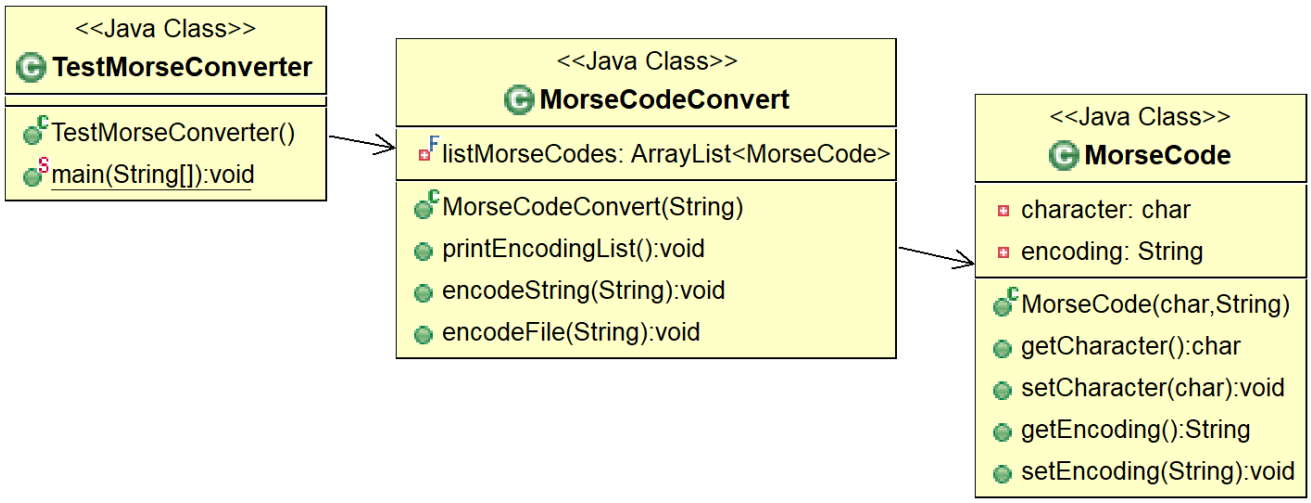

Figure 1: Assignment 3 Class Diagram