JAVA PROGRAMMING
(340)

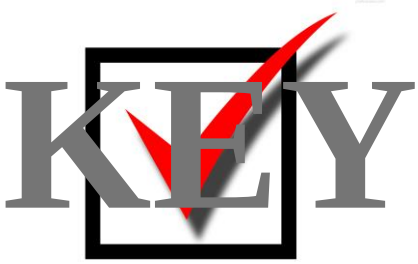
REGIONAL – 2015

Production Portion:

Program 1: ISBN Validation _____ (250 points)

TOTAL POINTS _____ ***(250 points)***

Judge/Graders: Please double check and verify all scores and answer keys!



ISBN Validation

XYZ, Inc., an international book reseller, sends and receives thousands of books each day. To track the incoming and outgoing books, inventory personnel scan each book's International Standard Book Number (ISBN) upon arrival. Occasionally, the barcode is damaged so that scanning it is impossible. In this situation, the inventory clerk simply writes down the ISBN number for manual entry at the end of the day. Unfortunately, the entry screen used for this end-of-day processing does not verify if the ISBN was entered correctly. Because of this, ISBN numbers are routinely entered into the system incorrectly. This causes problems in accounting when they attempt to set a price for the book and find that the ISBN number is invalid.

You have been requested to write a program that will verify the ISBN numbers in the system by calculating the check digit for each ISBN in the system. The input into your program is a file that contains ISBN numbers. Your program is to read the file of ISBN numbers and generate a report of the invalid ISBN numbers contained in the input file.

ISBN numbers are 10 digits long, including the check digit. The check digit is the last digit in an ISBN number. Each digit in the ISBN number has a weight associated with it. The first digit has a weight of 10, the second has a weight of 9, and the third has a weight of 8, etc., with the 10^{th} digit having a weight of 1.

To verify if an ISBN number is valid, the following algorithm is used. First, multiply each digit by its associated weight. Second, sum each of the products. Third, divide the sum of the products by 11. If the remainder is zero, then the ISBN is valid. For example, the process to verify 0-123-45678-9 is as follows.

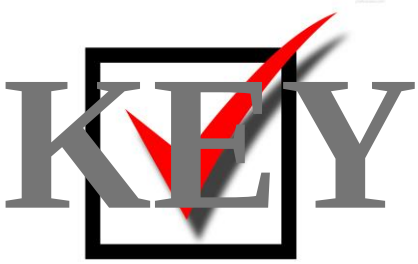
(0*10) + (1*9) + (2*8) + (3*7) + (4*6) + (5*5) + (6*4) + (7*3) + (8*2) + (9*1) = 165

165 / 11 = 15 (remainder of zero)

Note that if the check digit is 10, then the Roman numeral 'X' is used.

Requirements:

1. You must create an application named Java_340_ContestantNumber, where ContestantNumber is your BPA assigned contestant number (using the underscore in place of dashes). For example, Java_340_01_2345_6789.
2. Your name and contestant number must appear as a comment at the top of the main source code file.
3. The program must read the input file, "input340.txt" for the ISBN numbers to verify.
4. The program will calculate the check digit for each ISBN contained in the file and compare it with the current check digit.
5. The program must implement a method to determine if an ISBN number is valid.



6. The program must create an output file named, "output340.txt", which contains the invalid ISBN numbers from the input file.
7. The format of the output file will be same as the input file.
8. When the program executes, display a count of the records processed to the screen with a count of the invalid ISBN numbers detected.

You will have 90 minutes to complete your work.

Your name and/or school name should NOT appear on any work you submit for grading.

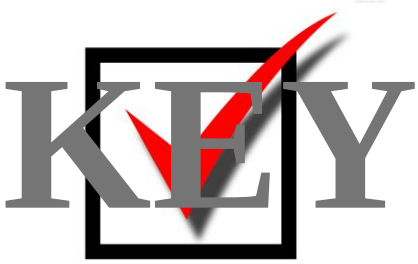
Copy your entire solution/project to the flash drive provided. You must submit your entire solution/project so that the graders may open your project to review the source code. You must ensure that the files required to run your program are present and will execute on the flash drive provided. Note that the flash drive letter may not be the same when the program is graded as it was when you created the program. It is recommended that you use relative paths rather than absolute paths to ensure that the program will run regardless of the flash drive letter. The graders **will not** compile or alter your source code to correct for this. Submissions that do not contain source code **will not** be graded.

Assumptions to make when taking this assessment:

- The input file(s) will contain only ASCII characters.
- A test input file will be available and will be named, "input330.txt".

Development Standards

- Standard name prefixes must be utilized for variables.
- All subroutines, functions, and methods must be documented with comments explaining the purpose of the method, the input parameters (if any), and the output (if any).



Your application will be graded on the following criteria:

Solution and Project

The project is present on the flash drive	____ 10 pts
The project is named according to the naming conventions	____ 10 pts

Program Execution

The program runs from the USB flash drive	____ 15 pts

If the program does not execute, then the remaining items in this section receive a score of zero.

The program runs and produces an output file	____ 20 pts
The program displays the correct count of the records processed (12)	____ 20 pts
The program displays the correct count of the invalid records detected (4)	____ 20 pts
The program correctly executes and reports the invalid ISBN numbers	____ 75 pts

The invalid ISBN numbers are: 1234-5678-9X, 0671-6571-53, 0152-0495-01, and 0153-0511-55.

Source Code Review

The source code is properly commented	
A comment containing the contestant number is present	____ 10 pts
Methods and code sections are commented	____ 20 pts
A method exists to perform the ISBN check	____ 30 pts
Code uses consistent variable naming conventions	____ 10 pts
Methods are used appropriately	____ 10 pts

Total Points: 250 pts

The input file contains 12 records and contains the following data.

1234-5678-9X
0123-4567-89
1234-5678-90
1843-5306-6X
0571-0898-95
0671-6571-53
1590-5930-06
0596-5299-45
0596-0072-48
0152-0495-01
0152-0494-1X
0153-0511-55

Four of the ISBN numbers are invalid.

The output file must contain the following invalid ISBN numbers. Order is not important; however, it is expected but not required that the file be processed from the top down.

1234-5678-9X
0671-6571-53
0152-0495-01
0153-0511-55