COMP20220 Programming II (Conversion)

Practical 4

Q1 Write a program to validate a credit card number using the *stepwise refinement* approach from Chapter 6 (Methods). Valid credit card numbers must satisfy the following criteria:

1) The number must have between 13 and 16 digits.
2) The number must have a valid prefix – i.e. it must start with any of the following:
   - 4 for Visa cards
   - 5 for Master cards
   - 37 for American Express cards
   - 6 for Discover cards
3) The number must satisfy the *Luhn check* (aka the *Mod 10 check*).

The Luhn check proceeds as follows:

(i) Double every second digit from right to left as per the example shown in Figure 1.

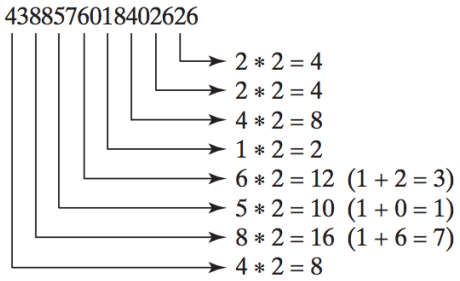

Figure 1: Luhn Check Example.

If doubling of a digit gives a two-digit number, add the two digits to get a single-digit number.

Add all single-digit numbers:

$$4 + 4 + 8 + 2 + 3 + 1 + 7 + 8 = 37$$

(ii) Add all digits in the odd places from right to left in the card number.

$$6 + 6 + 0 + 8 + 0 + 7 + 8 + 3 = 38$$

Sum the results from **(i)** and **(ii)** above:

$$37 + 38 = 75$$

If the sum is divisible by 10, the card number is valid; otherwise, it is invalid.

Write a program that prompts the user to enter a credit card number. Read in the number as a string. Display whether the number is valid or invalid. For example, the number 4388576018410707 is valid, but the number 4388576018402626 is not. Refer to the Design Diagram in Figure 2 and use the method headers specified below in your program.

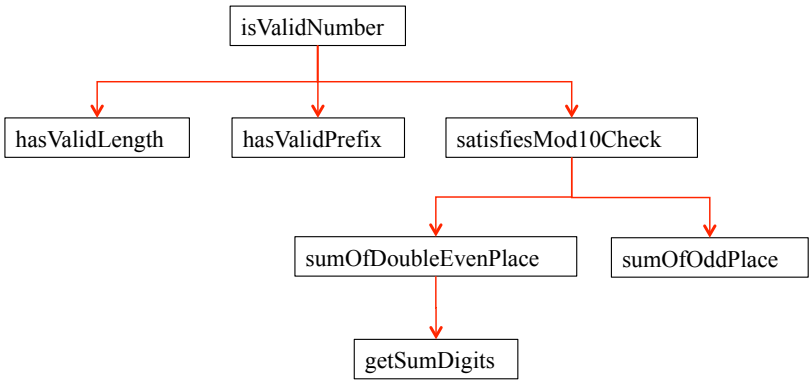

Figure 2: Design Diagram.

```
// Return true if the card number is valid
public static boolean isValidNumber(String number)

// Return true if the card number has between 13 and 16 digits
public static boolean hasValidLength(String number)

// Return true if the card number has a valid prefix
public static boolean hasValidPrefix(String number)

// Return true if the Mod 10 check is satisfied
public static boolean satisfiesMod10Check(String number)

// Double every second digit from right to left and return sum
public static int sumOfDoubleEvenPlace(String number)

// Return sum of digits in odd places from right to left
public static int sumOfOddPlace(String number)

// Return this number if it is a single digit;
// otherwise return the sum of the two digits
public static int getSumDigits(int number)
```

Q2 A string is a palindrome if its reversal is the same as itself. For example, the string `"abba"` is a palindrome, the string `"Abba"` is not and nor is the string `"Hello"`. Write a test program that prompts the user to enter a string and displays whether the string is a palindrome.

In your program, use methods with the following headers:

```
// Returns the reversal of a string; e.g.  reverse("Hello") returns "olleH"
public static String reverse(String s)

// Returns true if the string is a palindrome; false otherwise
public static boolean isPalindrome(String s)
```

Note: the method `isPalindrome` should call method `reverse`.

Q3 Write methods to compute the mean and standard deviation of a set of n numbers. The equations to compute these statistics are:

$$mean = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{x_1 + x_2 + \cdots + x_n}{n} \quad deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n - 1}}$$

Use the following method headers to calculate the mean and standard deviation:

```
 // Returns the mean of double values
public static double mean(double[] x)

// Returns the deviation of double values
public static double deviation(double[] x, double mean)
```

Write a test program that prompts the user to enter some numbers, stores the numbers in an array, and displays the mean and standard deviation.

Q4 Write a method that returns true if two integer arrays (`list1` and `list2`) have the same contents, but not necessarily in the same order. Use the following method header:

```
public static boolean contentEquals(int[] list1, int[] list2)
```

Write a test program that prompts the user to enter two lists of integers and displays whether the two have the same contents. Here are some sample runs. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list.

```
Enter list1: 5 2 5 6 6 1 Enter
Enter list2: 5 5 2 6 1 6 Enter
The two lists have the same content
```

```
Enter list1: 5 5 5 6 6 1 Enter
Enter list2: 5 2 5 6 1 6 Enter
The two lists do not have the same content
```

Hint:

(1) First check if the two arrays have the same size; if not, return false.

(2) Sort `list1` and `list2` using the `java.util.Arrays.sort` method. Then compare the corresponding elements from `list1` and `list2` and return true if all corresponding elements match.

Q5 Write the following method that tests whether an array contains (at least one instance of) four consecutive numbers with the same value:

```
public static boolean isConsecutiveFour(int[] values)
```

Write a test program that prompts the user to enter a series of integers and displays if the series contains four consecutive numbers with the same value. Here are some sample runs.

```
Enter the number of values: 8 Enter
Enter the values: 3 4 5 5 5 5 4 5 Enter
The list has consecutive fours
```

```
Enter the number of values: 9 Enter
Enter the values: 3 4 5 5 6 5 5 4 5 Enter
The list has no consecutive fours
```