COSC 600 Assignment 2
Due Oct 18th by 11:55pm (SK time)

General Instructions

- You will create a New Package called assign2SDCXXXXXX where XXXXXX is your assigned SDC Number. All Code for this assignment is to be placed in this package.
- Upon completion of this assignment you are to submit the source directory, either as is or in a zip file, to your submission directory.
- All Programs are to have appropriate comments identifying what the program is doing. Additionally the programs should include a comment at the beginning of each as per code guidelines and use informative variable names using camelCase, and otherwise follow good programing practice. A significant portion of the marks for the assignment will be for such considerations.
- This is an **individual assignment** – while students may discuss problems among themselves, all code is expected to be produced individually and will be checked thus.

Question 1: 10 Marks

The international standard letter/number mapping found on the telephone is shown below:

1	2 ABC	3 DEF
4 GHI	5 JKL	6 MNO
7 PQRS	8 TUV	9 WXYZ
	0	

Write a program, Q1, that reads a string and displays its corresponding telephone digit sequence. You may assume the following:

1. Use of 1 for spaces
2. There will be no special characters to handle.
3. All text to be processed is uppercase

<Output>

```
What string would you like processed? MY STRING
Phone #sequence: 691787464
```

<End Output>

Question 2: 10 Marks

A positive integer is a perfect number if it is equal to the sum of all its positive divisors. For example, six is the first perfect number as it's divisors are 3,2, and 1, whose sum is 6. The next is 28 whose divisors are 14,7,4,2,1.
There are exactly 4 perfect numbers between 1 and 10,000. Write a program, Q2, that determines and displays all these perfect numbers.

Question 3: 10 Marks

Write a program, Q3, which will ask the user to input a Base 13 number up to 5 digits and convert it to its equivalent decimal representation (note in Base 13 A is decimal 10, B is decimal 11, and C is decimal 12). Here are some sample runs that your program might produce:

```
<Output>
Enter a Base 13 number up to 5 digits: 4237A
The decimal number for 4237A is 119246
<End Output>
```

```
<Output>
Enter a Base 13 number up to 5 digits: 52A6
The decimal number for 52A6 is 11459
<End Output>
```

Question 4: 10 Marks

Star Trek episodes and movies have often used the concept of a star date to reflect a particular date (log entries). For the 2009 ***Star Trek*** Movie the writers used a System where they gave the current date followed by a period followed by the offset from the beginning of the year. Thus, in the movie the destruction of Vulcan occurred on 2258.42 or February 11, 2258 or the 42nd day of 2258. Likewise, Star Trek, "***The heart of Darkness"***, began on 2259.55 or February 24, 2259. So February 24 is the 55th day of 2259.

Write a program, Q4, that will ask for a Star Date and then convert it into the corresponding Calendar date. You may use the java DATE and/or CALENDAR classes for this question.

Question 5 10 Marks

(Dice Roll) Write a program Q5 that will simulate rolling a pair of dice 100,000 times, keeping track of the sum of the two die rolls. So for each roll of the dice, the value will be a total between 2 and 12. At the end of the program provide a report on how many times each of the combinations between 2 and 12 were rolled.

Question 6 10 Marks

Write a program, Q6, that will take in a line of input and report on the number of each of the vowels (aeiouy) that occur in the string. A report should be issued for each letter. You can assume all input is lowercase. Following is some sample output:

<Output>

```
Please enter a line of input to analyzed
This is a fantastic group of students.

Vowel count:

a    3
e    1
i    3
o    2
u    2
y    0
```

NOTE: The last three questions involve just creating methods. Create a single jUnit test file which contains ALL tests for all methods.

Question 7 10 Marks

Adapt your code from the first assignment for summing digits of a 4-digit number. You will do two things:

1. convert the code to a method which accepts a **single** int value and returns an int sum of its digits.
2. use looping to process any int value regardless of how many digits it has

```
public static int sumDigits(int inVal)
```

Question 8 10 Marks

A palindrome prime is a prime number that reads the same forwards or backwards. An example of a palindrome prime is 131. Write a method with the following signature for determining if a given number is a palindrome prime. You may use your prime number checker from class.

```
public static boolean isPallyPrime(int nVal)
```

Question 9 25 Marks

Write a program to validate credit card numbers. Credit card numbers follow certain patterns. A credit card number must have between 13 and 16 digits. It must start with
4 for Visa cards
5 for Mastercards
37 for American Express cards
6 for Discover cards

In 1954, Hans Luhn of IBM proposed an algorithm for validating credit card numbers. The algorithm is useful to determine whether a card number is entered correctly, or whether a credit card is scanned correctly by a scanner. Credit card numbers are generated following this validity check, commonly known as the Luhn check or the Mod 10 check, which can be described as follows :

for example, consider the card number 4388 5760 1840 2626:

1. First, the number starts with a 4 and is 16 digits long so it is valid as far as or preliminary checks are concerned.

Now the Luhn process:

2. Double every second digit from right to left. If doubling of a digit results in a two-digit number, add up the two digits to get a single-digit number.

   4388 5760 1840 2626

   2 * 2 = 4
   2 * 2 = 4
   4 * 2 = 8
   1 * 2 = 2
   6 * 2 = 12 (1 + 2 = 3)
   5 * 2 = 10 (1 + 0 = 1)
   8 * 2 = 16 (1 + 6 = 7)
   4 * 2 = 8

3. Now add all single-digit numbers from Step 1. 4+4+8+2+3+1+7+8 = 37
4. Add all digits in the odd places from right to left in the card number. 6+6+0+8+0+7+8+3 = 38
5. Sum the results from Step 2 and Step 3. 37 + 38 = 75

6. If the result from Step 4 is divisible by 10, the card number is valid; otherwise, it is invalid. The number above: 4388 5760 1840 2626 is invalid, but another number: 4388 5760 1841 0707 is valid.

Write a program that prompts the user to enter a credit card number as a long integer. Display whether the number is valid or invalid. Design your program to use the following methods:

```
/** Return true if the card number is valid */
   public static boolean isValid(long number)

/** Get the result from Step 2 */
   public static int sumOfDoubleEvenPlaces(long number)

/** Return number if it is a single digit, otherwise,
  * return the sum of the two digits(from step 1) */
   public static int getSumDigits(int number)

/** Return sum of odd-place digits in number (step 3)*/
   public static int sumOfOddPlaces(long number)

/** Return true if number has a proper prefix */
   public static boolean prefixCorrect(long number, int d)

/** Return the number of digits in d */
   public static int getSize(long d)
```

Here are sample runs of the program:

```
Enter a credit card number as a long integer:
4388576018410707
4388576018410707 is valid

Enter a credit card number as a long integer:
4388576018402626
4388576018402626 is invalid
```