CEGEP VANIER COLLEGE
Department of Computer Science

420-101-VA PROGRAMMING 1 sections 1 & 2

Assignment_04: Due November 30th, 2023, at 11:59pm **Prof. Nagat Drawel**

Overview:

The purpose of this assignment is to learn:

1. More on String and Character classes
2. Wrapper Classes
3. A first look at user-defined classes

Notes:

1. Please make sure to follow the programming standards (e.g., naming conventions, comments, @author); Keep your code clean (Indentation, Spaces, etc.,) not doing so will result in loss of marks.
2. Please demonstrate your working programs to your teacher. Failing to explain your own code will be heavily panelized.
3. Please submit one .java file for each task (3 java files in total, Omnivox accepts multi-submission). DO NOT ZIP YOUR FILES.

Task 01

1. Write a class named `stringArrayTools` that has no instance variables and includes the following static methods:
    a. `mostFrequent`. This method accepts a reference to a String object as an argument and returns the character that occurs the most frequently in the object.
    b. `toTitlecase`: This method accepts a reference to a String object and converts a string to title case (the first letter is upper case, and the rest of the string are lower case.) (This method must be used for loop; the substring () method in the String class is not allowed.)
    c. `toCamelcase`: This method accepts a reference to a String object and converts a string to camel case (remove all the spaces and bring the next character to uppercase. The first character of the whole string should also be uppercase.) (This method must use for loop; the substring () method in the String class is not allowed)
    d. `countStringScore`: This method accepts a reference to a String object and counts a score of a string. Hint:
        i. Each English character worth a score of 2
        ii. Each space worth a score of 03
        iii. Each digit worth a score of 1
        iv. Each other symbol worth a score of 3
        v. The final score of a string is the total score divided by the length of the string

vi. null strings and empty strings have a score of 0

Task 02

1. Create a project Department, then inside of the package
2. Create a class **Course**, which contains:
    a. Data members:
        i. String courseName
        ii. int credit
        iii. boolean isComplimentaryCourse
    b. Methods:
        i. Default constructor: empty string for courseName, 0 for credit and false for isComplimentaryCourse
        ii. Constructor that takes courseName: 0 for credit and false for isComplimentaryCourse
        iii. Constructor with all three data members
        iv. toString: print a course with the following

```
Course Name  : Programming I
Credit       : 3
Complimentary: No
```

        v. equals
        vi. Getter and setter

Task 3:

Write a class **Calendar**, the class should contain:

a. Data members:
    a. int year
    b. int month
    c. int day
b. Methods:
    i. Default constructor // 2023, 1, 1 for year, month and day
    ii. Constructor with year, month and day
    iii. Copy Constructor
    iv. `increaseDay()`: to increase the day by 1, when it reaches the last day of a month, the day go back to 1 and the month gets increased.
    v. `increaseMonth()`: to increase the month by 1, when it reaches the last month of a year, the month go back to 1 and the year gets increased.
    vi. `increaseYear()`: to increase the year by 1.
    vii. `isLeapYear()`: to check if a year is a leap year

```
if (year is not divisible by 4) then (it is a common year)
    else if (year is not divisible by 100) then (it is a leap year)
    else if (year is not divisible by 400) then (it is a common year)
    else (it is a leap year)
```

vii. `getDaysInMonth`(): to check how many days are there in the current month. E.g.: if it is Jan, then 31, if it is Sep, then 30, if it is Feb, non-leap year, then 28, if it is Feb, leap year, then 29.
viii. `toString`()
ix. `equals`()
x. getter and setter.