

stanCode

標準程式教育機構

Assignment 4



這份作業將幫助同學們更熟悉**資料結構**的運用。在這個過程中，你將學習如何使用 **array**、**arraylist** 和 **hashmap** 來處理各種不同的資料。

如果作業卡關 **歡迎與助教討論**，stanCode 也非常鼓勵同學們互相討論作業的概念，但 **請不要把自己的 code 給任何人看**，分享您的 code 會剝奪其他學生獨立思考的機會，同時會導致其他學生的程式碼與您的 code 極度相似，使得防抄襲軟體認定有抄襲之嫌疑。

Problem 1 - ReviewAnalysis.java

在每次造訪網站結束時，網站的使用者會被要求提供對該網站的評論。每個評論由 **Review** class 中的 object 表示，其中包含一個整數，表示使用者對網站的評分，以及一個可選的 **String** 評論。Review object 中的評論以句號(“.”)、驚嘆號(“!”)、或字母結尾，而如果使用者未輸入評論，則為長度為 0 的 **String**。

```
public class Review
{
    private int rating;
    private String comment;

    /** Precondition: r >= 0
     *      c is not null.
     */
    public Review(int r, String c)
    {
        rating = r;
        comment = c;
    }

    public int getRating()
    {
        return rating;
    }

    public String getComment()
    {
        return comment;
    }

    // There may be instance variables, constructors, and methods that are not shown.
}
```

ReviewAnalysis class 包含用於分析使用者提供的評論的 **method**。您需要在 **ReviewAnalysis** class 中撰寫兩個 **method**。

```
public class ReviewAnalysis
{
    /** All user reviews to be included in this analysis */
    private Review[] allReviews;

    /** Initializes allReviews to contain all the Review objects to be analyzed */
    public ReviewAnalysis()
    { /* implementation not shown */ }

    /** Returns a double representing the average rating of all the Review objects to be
     * analyzed, as described in part (a)
     * Precondition: allReviews contains at least one Review.
     * No element of allReviews is null.
     */
    public double getAverageRating()
    { /* to be implemented in part (a) */ }

    /** Returns an ArrayList of String objects containing formatted versions of
     * selected user comments, as described in part (b)
     * Precondition: allReviews contains at least one Review.
     * No element of allReviews is null.
     * Postcondition: allReviews is unchanged.
     */
    public ArrayList<String> collectComments()
    { /* to be implemented in part (b) */ }
}
```

(a) 請撰寫 **ReviewAnalysis** class的 **getAverageRating** method，該method會回傳 **allReviews** 中所有元素的平均評分（算術平均）。例如，如果 **allReviews** 包含以下 **Review** object，則 **getAverageRating** 將回傳 **3.4**。

0	1	2	3	4
4 "Good! Thx"	3 "OK site"	5 "Great!"	2 "Poor! Bad."	3 ""

請完成 **getAverageRating** method。

```
/** Returns a double representing the average rating of all the Review objects to be
 * analyzed, as described in part (a)
 * Precondition: allReviews contains at least one Review.
 * No element of allReviews is null.
 */
public double getAverageRating()
```

(b) 請撰寫 **ReviewAnalysis** class的 **collectComments** method，該 **method** 僅收集包含驚嘆號的評論，並將其格式化，並回傳一個包含使用者評論的 **String** objects 之 **ArrayList**。這些評論來自於 **allReviews**，且包含驚嘆號，格式如下所示。如果 **allReviews** 中沒有任何評論包含驚嘆號，則回傳一個空的 **ArrayList**。

- 回傳 **ArrayList** 中的 **String** 以在 **allReviews** 中的 **Review** 的索引開頭
- 索引後面緊接著一個連字符號 ("-")
- 連字符號後面是原始評論的副本
- **String** 必須以句號或驚嘆號結尾；如果 **allReviews** 中的原始評論沒有以句號或驚嘆號結尾，則添加一個句號

以下 **allReviews** 的例子與(a)部分之例子相同：

0	1	2	3	4
4 "Good! Thx"	3 "OK site"	5 "Great!"	2 "Poor! Bad."	3 ""

使用給定的 **allReviews** 內容呼叫 **collectComments** 將回傳以下的**ArrayList**。在 **allReviews** 中，索引為 **1** 和索引為 **4** 的評論不包含驚嘆號，因此它們不會包含在回傳的 **ArrayList** 中。

"0-Good! Thx."	"2-Great!"	"3-Poor! Bad."
----------------	------------	----------------

請完成 **collectComments** method。

```
/** Returns an ArrayList of String objects containing formatted versions of
 * selected user comments, as described in part (b)
 * Precondition: allReviews contains at least one Review.
 * No element of allReviews is null.
 * Postcondition: allReviews is unchanged.
 */
public ArrayList<String> collectComments()
```

Problem 2 - ArrayResizer.java

這個問題涉及到處理一個包含整數的二維陣列。您將重新撰寫 **ArrayResizer** class 的兩個 static method，如下所示：

```
public class ArrayResizer
{
    /** Returns true if and only if every value in row r of array2D is non-zero.
     * Precondition: r is a valid row index in array2D.
     * Postcondition: array2D is unchanged.
     */
    public static boolean isNonZeroRow(int[][] array2D, int r)
    { /* to be implemented in part (a) */ }

    /** Returns the number of rows in array2D that contain all non-zero values.
     * Postcondition: array2D is unchanged.
     */
    public static int numNonZeroRows(int[][] array2D)
    { /* implementation not shown */ }

    /** Returns a new, possibly smaller, two-dimensional array that contains only rows
     * from array2D with no zeros, as described in part (b).
     * Precondition: array2D contains at least one column and at least one row with no zeros.
     * Postcondition: array2D is unchanged.
     */
    public static int[][] resize(int[][] array2D)
    { /* to be implemented in part (b) */ }
}
```

(a) 請撰寫 **isNonZeroRow** method，且僅當二維陣列 **array2D** 中第 **r** 行的所有元素都不等於零時，回傳 **true**。

例如，當初始化了一個二維陣列：

```
int[][] arr = {{2, 1, 0},
               {1, 3, 2},
               {0, 0, 0},
               {4, 5, 6}};
```

則以下是對 **isNonZeroRow** 呼叫的範例：

呼叫 isNonZeroRow	回傳值	說明
ArrayResizer.isNonZeroRow(arr, 0)	false	第 0 行中至少有一個值為零
ArrayResizer.isNonZeroRow(arr, 1)	true	第 1 行中的所有值皆不為零
ArrayResizer.isNonZeroRow(arr, 2)	false	第 2 行中至少有一個值為零
ArrayResizer.isNonZeroRow(arr, 3)	true	第 3 行中的所有值皆不為零

請完成 **isNonZeroRow** method。

```
/** Returns true if and only if every value in row r of array2D is non-zero.
 * Precondition: r is a valid row index in array2D.
 * Postcondition: array2D is unchanged.
 */
public static boolean isNonZeroRow(int[][] array2D, int r)
```

(b) 請撰寫 **resize** method，該 method 會回傳一個新的二維陣列，其僅包含 **array2D** 中所有元素都為非零值的行。新陣列中的元素應按照它們在原始陣列中出現的順序出現。

以下程式碼片段初始化了一個二維陣列並呼叫了 **resize** method：

```
int[][] arr = {{2, 1, 0},
               {1, 3, 2},
               {0, 0, 0},
               {4, 5, 6}};
int[][] smaller = ArrayResizer.resize(arr);
```

且當程式碼片段完成後，**smaller** 的內容如下所示：

```
{{1, 3, 2}, {4, 5, 6}}
```

另外，一個名為 **numNonZeroRows** 的輔助 method 已經提供給您。該 method 會回傳其二維陣列參數中不包含零值的行數。

請完成 **resize** method，並假設 **isNonZeroRow** method 按說明執行，且先不論您在(a)部分寫的內容。您必須適當使用 **numNonZeroRows** 和 **isNonZeroRow** method 才能獲得全部的分數。

```
/** Returns a new, possibly smaller, two-dimensional array that contains only rows from array2D
 * with no zeros, as described in part (b).
 * Precondition: array2D contains at least one column and at least one row with no zeros.
 * Postcondition: array2D is unchanged.
 */
public static int[][] resize(int[][] array2D)
```


Problem 3 - WordPair.java

這個問題涉及到由以下 **WordPair** class 表示之詞對 (pairs of words) :

```
public class WordPair
{
    /** Constructs a WordPair object. */
    public WordPair(String first, String second)
    { /* implementation not shown */ }

    /** Returns the first string of this WordPair object. */
    public String getFirst()
    { /* implementation not shown */ }

    /** Returns the second string of this WordPair object. */
    public String getSecond()
    { /* implementation not shown */ }
}
```

您將為以下的 **WordPairList** class 撰寫 constructor 和另一個 method :

```
public class WordPairList
{
    /** The list of word pairs, initialized by the constructor. */
    private ArrayList<WordPair> allPairs;

    /** Constructs a WordPairList object as described in part (a).
     * Precondition: words.length >= 2
     */
    public WordPairList(String[] words)
    { /* to be implemented in part (a) */ }

    /** Returns the number of matches as described in part (b).
     */
    public int numMatches()
    { /* to be implemented in part (b) */ }
}
```

(a) 請為 **WordPairList** class 撰寫 constructor。其以一個字串陣列 **words** 作為參數，並將 instance variable **allPairs** 初始化為一個 **WordPair** objects 的 **ArrayList**。

一個 **WordPair** object 由陣列中的一個詞，與陣列中後面出現的一個詞組成。**allPairs** 列表包含了每個 **i** 和 **j** 的 **WordPair** objects (**words[i]**, **words[j]**)，其中 **0 ≤ i < j < words.length**。而**每個 WordPair object 只會被添加一次到列表中**。

以下的例子表示了兩個不同的 **WordPairList** objects：

Example1：

```
String[] wordNums = {"one", "two", "three"};
WordPairList exampleOne = new WordPairList(wordNums);
```

在以上程式碼執行後，**exampleOne** 的 **allPairs** instance variable 將以某種順序包含以下的 **WordPair** objects。

("one", "two"), ("one", "three"), ("two", "three")

Example2：

```
String[] phrase = {"the", "more", "the", "merrier"};
WordPairList exampleTwo = new WordPairList(phrase);
```

在以上程式碼執行後，**exampleTwo** 的 **allPairs** instance variable 將以某種順序包含以下的 **WordPair** objects。

("the", "more"), ("the", "the"), ("the", "merrier"),
("more", "the"), ("more", "merrier"), ("the", "merrier")

(b) 請撰寫 **WordPairList** class 的 **numMatches** method。該 **method** 會回傳 **allPairs** 中的 **WordPair object** 中兩個字串匹配的數量。

例如，下方的程式碼創建了一個 **WordPairList** object：

```
String[] moreWords = {"the", "red", "fox", "the", "red"};  
WordPairList exampleThree = new WordPairList(moreWords);
```

在程式碼執行後，**exampleThree** 的 **allPairs** instance variable 將以某種順序包含以下的 **WordPair** object。其中，第一個字串與第二個字串匹配的詞對 (pairs of words) 已被標記以便說明。

```
("the", "red"), ("the", "fox"), ("the", "the"),  
("the", "red"), ("red", "fox"), ("red", "the"),  
("red", "red"), ("fox", "the"), ("fox", "red"),  
("the", "red")
```

則若呼叫 **exampleThree.numMatches()** 應回傳 2。

Problem 4 (HashMap) - Anagram.java

給定兩個字串 **s** 和 **t**，如果 **t** 是 **s** 的易位構詞 (anagram)，回傳 **true**；若否，則回傳 **false**。

易位構詞 (Anagram)，是指透過重新排列不同字詞或片語的字母，以形成一個新的字詞或片語，通常**要使用所有在原本子詞或片語的字母，而且每個字母只使用一次**。

Example 1 :

Input : **s = 'anagram' , t = 'nagaram'**

Output : **true**

Example 2 :

Input : **s = 'rat' , t = 'car'**

Output : **false**

限制 :

- **1 <= s.length , t.length <= 5 * 10⁴**
- **s 和 t 皆由小寫的英文字母組成**

請撰寫一個名為 **Anagram** 的 class，其包含一個 static method **public static boolean checkAnagram(String s, String t)**，並 **請確保您使用 HashMap 來解決這個問題**。

Hint :

比較兩個 HashMap 是否相同，可以使用 : hashMap1.equals(hashMap2)

C/C++ 作業講解影片

以下提供 C/C++ 版本的作業講解影片，透過觀看，同學們可以輕鬆地轉換至不同的語言環境，同時深入瞭解不同語言作業所需的程式概念和語法。相信將為同學提供更廣泛的學習視野。

影片連結：<https://youtu.be/QkuwPuqykWI>

希望這份資源對同學們的學習旅程有所助益，並取得更好的學習成果！