

משימה #7 בקורס תוכנה 1

חלק א': החווה של מקדונלד הזקן

Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some chicks, E-I-E-I-O
With a cluck-cluck here and a cluck-cluck there
Here a cluck there a cluck
Everywhere a cluck-cluck
Old MacDonald had a farm, E-I-E-I-O

Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some cows, E-I-E-I-O
With a moo-moo here and a moo-moo there
Here a moo there a moo
Everywhere a moo-moo
With a cluck-cluck here and a cluck-cluck there
Here a cluck there a cluck
Everywhere a cluck-cluck
Old MacDonald had a farm, E-I-E-I-O

Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some dogs, E-I-E-I-O
With a woof-woof here and a woof-woof there
Here a woof there a woof
Everywhere a woof-woof
With a moo-moo here and a moo-moo there
Here a moo there a moo
Everywhere a moo-moo
With a cluck-cluck here and a cluck-cluck there
Here a cluck there a cluck
Everywhere a cluck-cluck
Old MacDonald had a farm, E-I-E-I-O

Requirements:

In Old MacDonald's farm you can find: dogs, cows, pigs, chicks and horses. In this exercise you will write an application that receives as an input a list of animals in old MacDonald's farm (with possible repetitions). The application prints:

1. **The list of animals in old MacDonald's farm with their sounds.** The order of the animals in this list is exactly the order in the input list. For example: if the input was "cow pig chick chick cow" then the output is

```
cow: moo
pig: oink
chick: cluck
chick: cluck
cow: moo
```

2. **The status of old MacDonald's farm:** a table of two columns where the first column contains animal names (no repetitions!!!) in alphabetical order, and the second columns contains the number of animals of this type in old MacDonald's farm. For example: if the input was "cow pig chick chick cow" then the output is:

Animal	Count
chick	2
cow	2
pig	1

3. **The "old MacDonald's had a farm" song for the animals in the farm.** For every animal the line "*And on his farm he had some ...*" appears at most once. The order of appearance of the animals in the song is as the order of appearance in the input list. For example: if the input was "cow pig chick chick cow" then the output is:

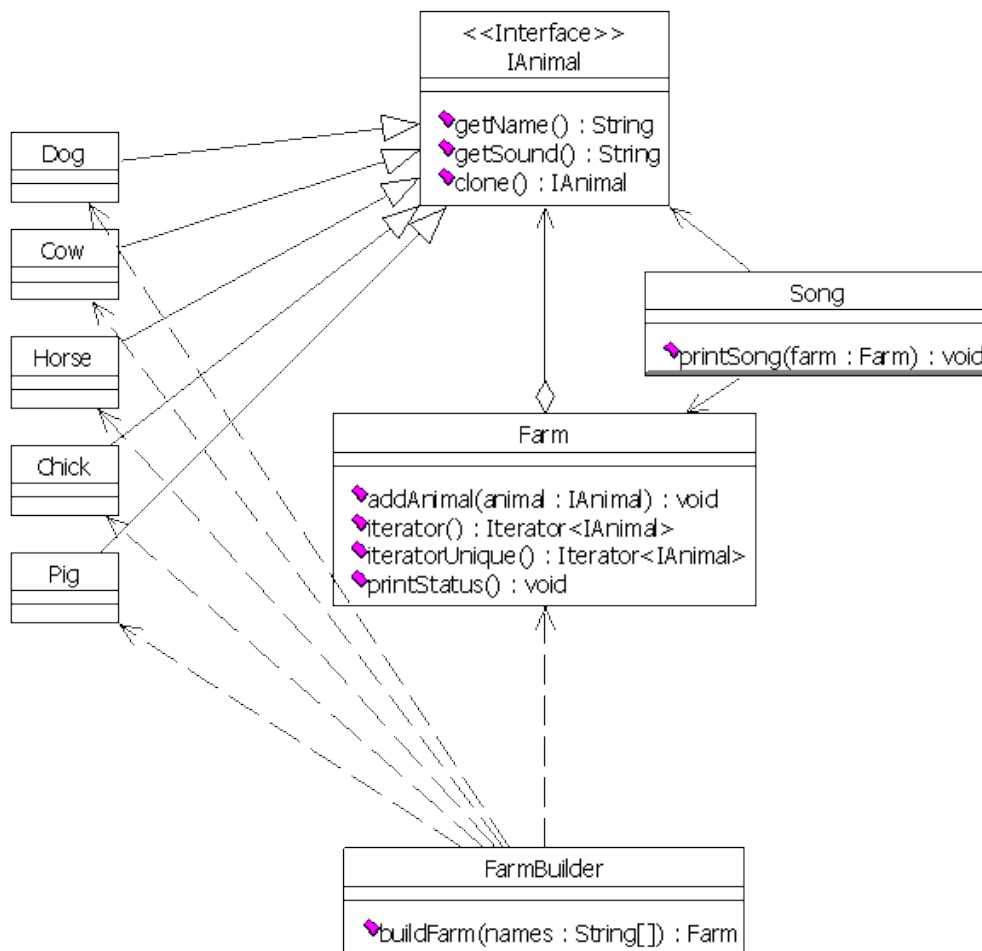
```
Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some cows, E-I-E-I-O
With a moo-moo here and a moo-moo there
Here a moo there a moo
Everywhere a moo-moo
Old MacDonald had a farm, E-I-E-I-O
```

```
Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some pigs, E-I-E-I-O
With an oink-oink here and an oink-oink there
Here an oink there an oink
Everywhere an oink-oink
With a moo-moo here and a moo-moo there
Here a moo there a moo
```

Everywhere a moo-moo
Old MacDonald had a farm, E-I-E-I-O

Old MacDonald had a farm, E-I-E-I-O
And on his farm he had some chicks, E-I-E-I-O
With a cluck-cluck here and a cluck-cluck there
Here a cluck there a cluck
Everywhere a cluck-cluck
With an oink-oink here and an oink-oink there
Here an oink there an oink
Everywhere an oink-oink
With a moo-moo here and a moo-moo there
Here a moo there a moo
Everywhere a moo-moo
Old MacDonald had a farm, E-I-E-I-O

Design:



The classes: Farm, FarmBuilder and Song belong to package il.ac.tau.software1.oldmac. The interface IAnimal and the classes implementing it (Pig, Cow, Horse, Chick and Dog) all belong to package il.ac.tau.software1.oldmac.animals.

The class Main (not shown in the diagram) is the entry point to the application (i.e. its main method should be run). Main and all the classes in il.ac.tau.software1.oldmac.animals are implemented and should not be changed.

The skeleton for the "Old MacDonald's farm" application is available on the course's web site (oldmac.zip)

You are required to complete the implementation of the classes FarmBuilder, Farm and Song as described below:

FarmBuilder:

- `public Farm buildFarm(String[] animalNames)`

Receives a list of animal types, builds and returns a new Farm object with those animals.

Farm:

- `public void addAnimal(IAnimal animal)`

Add a new animal to the farm.

- `public Iterator<IAnimal> iterator()`

Return an iterator of all the animal in the farm. The ordering is determined by the order of their addition to the farm.

- `public Iterator<IAnimal> iteratorUnique()`

Returns an iterator of the animals in the farm **without repetitions**. The iterator iterates the animals in the farm by the order of their addition to the farm. For example, if the animals added to the farm were: cow, pig, chick, chick, cow (in this order), then the iterator will iterate cow, pig, chick (in this order).

- `public void printStatus()`

Prints the status of the farm as described in the second requirement

Song:

- `public static void printSong(Farm farm)`

Prints the "Old MacDonald had a farm" song as described in the third requirement.

You may add any methods and fields you deem necessary to those three classes (do not change any other class). In your implementation you should use classes (and interfaces) from the Java framework collections.

You may assume that:

- The list of arguments to the application is not empty and that every argument is one of the following: "cow", "chick", "horse", "dog" or "pig".
- The method `Iterator.remove()` is never called for the two iterators of class `Farm`.

חלק ב' : חפש אותי

בחלק זה אתם נדרשים לממש מנוע חיפוש פשוט. [חלקים מהקוד](#) כבר נכתבו עבורכם והם זמינים להורדה באתר הקורס.

מנוע החיפוש שלנו יטפל במספר מצומצם של דפי HTML אותם הוא יקרא מהרשת. דפים אלו קבועים מראש. תוכלו למצוא את הרשימה בקובץ Main.java. אם תרצו תוכלו להוסיף או לשנות את הדפים בהם אתם משתמשים. לאחר שהורדנו דף HTML מהרשת נתייחס רק לחלק הטקסט שבדף ונפרק חלק זה למילים בודדות. חלקים אלו כבר נכתבו במחלקה HTMLTokenizer. בנוסף המחלקה Main שבה הקוד המפעיל את המערכת ומתקשר עם המשתמש קיימת אף היא.

מה עליכם לעשות:

נרצה ליצור אינדקס של כל המילים שהופיעו בכל הדפים שהורדנו מהרשת. קיומו של אינדקס זה יאפשר לנו מאוחר יותר לבצע חיפושים עבור מילה מסוימת. עליכם לממש את המחלקה WordIndex. מחלקה זו שומרת את אינדקס המילים, מאפשרת הוספת מילים לאינדקס וחיפוש בו.

```
public class WordIndex {  
  
    public WordIndex() {  
        ...  
    }  
  
    /**  
     * Add the words originating in the specifies URL.  
     * @param words - collection of words to add  
     * @param strURL - the location of the page containing the words  
     */  
    public void index(Collection<String> words, String strURL) {  
        ...  
    }  
  
    /**  
     * Search for a given word in the index  
     * @param word - the word to search  
     * @return A list of pages containing the word. The pages are  
     *         ordered according to the relative importance of the word  
     *         within them.  
     */  
    public List<String> search(String word) {  
        ...  
    }  
  
    ...  
}
```

• **המתודה index**

מתודה זו אחראית על אכלוס מבנה הנתונים שלכם. המתודה מקבלת אוסף של מילים (ייתכנו חזרות) ואת כתובת האינטרנט של הדף מהן הגיעו. עליכם לבחור את מבני הנתונים שבהם תשתמשו ולדאוג לשמור על הקשרים הבאים: לכל מילה באילו דפים היא מופיעה וכמה פעמים בכל דף, לכל דף כמה מילים בשה"כ מופיעות בו (עם חזרות). רשימת המילים בקלט היא כפי שהופיעה בדף המקורי, אולם יש לשמור גרסת lowercase של המילים.

• המתודה search

מקבלת מילה kjhpua ומחזירה רשימה ממוינת של כתובות אינטרנט בהן המילה מופיעה. נמיון את הרשימה כך שכל שהמשקל היחסי של מילה בדף גבוה יותר כך הכתובת תופיע במקום גבוה יותר ברשימה. את המשקל היחסי של מילה בדף נחשב לפי מספר המופעים של המילה בדף מחולק במספר המילים הכולל באותו דף.

הדרכה:

השתמשו במחלקות מ Java Collections, בפרט המנשק Map והמחלקה HashMap יכולים להועיל.

את מיון הרשימה של כתובות האינטרנט בצעו בעזרת הפונקציה Collections.sort(...). שימו לב שה"מיון הטבעי" של הכתובות (String) הוא מיון לקסיקוגרפי ולא כפי שמתבקש בשאלה. כדי לשנות את שיטת המיון עליכם לכתוב מחלקה המממשת את המנשק Comparator. כדי להקל את המימוש כתבו מחלקה זו כמחלקה פנימית במחלקה Wordindexer (זו המלצה בלבד). שימו לב שתוצאות ההשוואה תלויות במלת החיפוש הנוכחית.

דוגמאות:

עבור רשימת הכתובות המופיעה בקובץ ה Main ומילת החיפוש "java" יתקבל הפלט:

```
> java
1. http://www.gutenberg.org/files/27152/27152-h/27152-h.htm
2. http://en.wikipedia.org/wiki/Java
3. http://en.wikipedia.org/wiki/Java_(programming_language)
4. http://www.java.com/en/
```

הוראות הגשה:

1. קראו בעיון את [קובץ נוהלי הגשת התרגילים](#) אשר נמצא באתר הקורס.
2. הגשת התרגיל תעשה ע"י המערכת VirtualTAU (<http://virtual.tau.ac.il>). הוראות שימוש במערכת ניתן למצוא ב-<http://virtual2002.tau.ac.il/upload/misc/main1.html>
3. הגשת התרגיל תתבצע ע"י יצירת קובץ zip שנושא את שם המשתמש. לדוגמא, עבור המשתמש zvainer יקרא הקובץ zvainer.zip.
קובץ ה zip יכיל:
 - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז. זהות שלכם.
 - ב. קבצי ה-java של התכניות שהתבקשתם לכתוב.
 - ג. קובץ טקסט עם העתק של כל קבצי ה Java.