Advanced Programming – Assignment 1

Easy operations on sets of identifiers

Important notice up front

When necessary, use the following methods for the assignments below.

N.B. For the following methods to work, the delimiter of the Scanner-object has to be set to the empty String by calling the useDelimiter() method.

```
// Method to read 1 character.
char nextChar (Scanner in) {
    return in.next().charAt(0);
}
```

The three methods below have in common that they do not read a character from input, but return a `boolean` indicating whether a specific character could be read with `nextChar()`.

```
// Method to check if the next character to be read when
// calling nextChar() is equal to the provided character.

boolean nextCharIs(Scanner in, char c) {
    return in.hasNext(Pattern.quote(c+""));
}
```

To be able to use the class `Pattern`, include the following import in your program:

```
import java.util.regex.Pattern;

// Method to check if the next character to be read when
// calling nextChar() is a digit.

boolean nextCharIsDigit (Scanner in) {
    return in.hasNext("[0-9]");
}

// Method to check if the next character to be read when
// calling nextChar() is a letter.

boolean nextCharIsLetter (Scanner in) {
    return in.hasNext("[a-zA-Z]");
}
```

Description assignment

An interactive program is a program that has interaction between the user and the program. Concretely: the program asks questions and the user gives answers, after which the program processes these answers and prints the results.

For this assignment a program must be written that reads two sets of identifiers from standard input, applies four operations on these sets, and prints the results of these operations in standard output.

Details of identifiers

The **identifiers**, which form the elements of the sets used in this assignment, must have the following properties:

- Only alphanumeric characters are allowed as elements of an identifier.
- An identifier begins with a letter.
- Identifiers have a length of at least 1 character.

Details of sets

The **sets**, used in this assignment, have to meet to following properties:

- Only identifiers, following the restrictions used in this assignments, are allowed as elements of a set.
- sets can contain a minimum of 0 and a maximum of 20 identifiers.

The four **operations** on sets are defined as follows:

1. *difference*: all elements contained in the 1st but not the 2nd set.
2. *intersection*: all elements contained in both sets.
3. *union*: all elements of both sets. (N.B. sets do not contain duplicate elements per definition.)
4. *symmetric difference*: all elements of both sets that are not contained in the intersection.

Details of program behaviour

Input to the program is through standard input (the keyboard) and output uses standard output (the screen). The program asks two times for a set of identifiers and subsequently computes the difference, intersection, union, and symmetric difference. The program is supposed to repeat the process described above until CTRL-D (or in the case of Windows: CTRL-Z) is entered as input to one of the questions of the program. E.g. :

```
Give the first set : {ape nut mouse mouse2}
Give the second set : {eel w xyz nut}
difference = {ape mouse mouse2}
intersection = {nut}
union = {ape nut mouse2 eel w xyz mouse}
sym. diff. = {ape mouse eel w xyz mouse2}

Give the first set : {one two three}
Give the second set : {}
difference = {one two three}
intersection = {}
union = {one two three}
sym. diff. = {one two three}

Give the first set :
Give the first set : {peter jan hank
Missing '}'
Give the first set : {peter jan hank}
Give the second set : ^D
```

Note that the text following the colon was typed by the user by pressing CTRL-D.

sets on the input can contain a maximum of only 10 identifiers, to ensure that it is always possible to determine the union and symmetric difference. The identifiers in a set on the input are delimited by spaces and are preceded and closed by respectively { and }. Because sets are unordered by definition, there are no restrictions on the order of identifiers in the input and output of a set. On improper input, a clear error message has to be given in one line. Absence of input is not seen as an error, but should lead to a repeat of the question.