German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher

Data Structures and Algorithms, Winter term 2020
Practice Assignment 4

Exercise 4-1 Average GPA

Given a stack of `Student` objects you are required to implement a method that calculates the average `GPA` of some students given in a stack (`StackObj`) and returns this average value.

Exercise 4-2 Infix to Postfix

Write a static method `String infixToPostfix(String infix)` that takes for input a string representing an infix expression and returns the same expression in postfix representation. For example, the infix expression `"1+2*3+4"` will return the expression `"123*+4+"`. The input string will only consists of the digits '0' to '9' and the '*' and '+' operators only. Make use of a stack to solve this problem.

Exercise 4-3 Browsing History

You can protect your privacy by editing or completely erasing your browsing history. A browsing list consists of collection of links. A link can be defined as follows:

```
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;

class Link {
    String url;
    String date; // Format YYYY-MM-DD
    int frequency; // number of times a url has been visited

    public Link(String url) {
        this.url = url;
        this.frequency = 0;

        // gets the current date
        Calendar cal = Calendar.getInstance();
        Date calDate = cal.getTime();
        // and convert it into YYYY-MM-DD
        SimpleDateFormat format1 = new SimpleDateFormat("yyyy-MM-dd");
        try{
            date= format1.format(calDate);
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }

    public Link(String url, int frequency) {
        this(url);
        this.frequency = frequency;
    }

    public String toString(){
        return url +"␣" + date+"␣frequency␣"+frequency;
    }
}
```

Your task is to implement a datatype for the browsing history with the following instance methods:

a) Adding a specific URL to the current browsing history. This operation should be done in O(n).

```
void add(String url)
```

b) Removing the latest URL you have visited. This operation should be done in O(1).

```
void removeLast()
```

c) Removing the history of a specific day. This operation should be done in O(n).

```
void removeLast(String date)
```

d) Return the total number of links you have in your browsing list. This operation should be done in O(1).

```
int getNumberOfLinks()
```

e) Search for the most viewed link in your browsing list. This operation should be done in O(n).

```
String search()
```

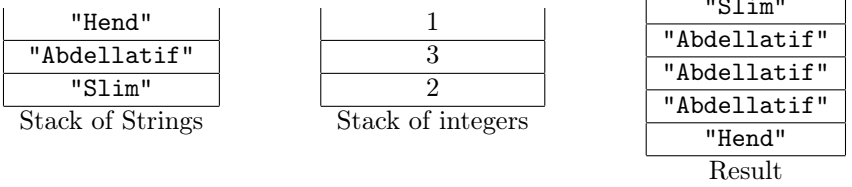
Use the best datatype to ensure the complexity listed above for all methods. You are only allowed to use one type of data structure and once you decided about the data structure, you are only allowed to use two instances of this type.

You should use one of the datatypes discussed in class. However, you are not allowed to use neither arrays nor strings.

Exercise 4-4 Duplicating Elements of a Stack

Write a Java method `createDuplicates` that takes 2 stacks of equal size as input. The first one is a stack of strings and the second one is a stack of integers. The method should return a new stack of strings where each element of the original string stack is repeated the same number of times specified in the corresponding cell in the integer array.

For example:



Exercise 4-5 Palindrome using Stacks
To be discussed in the Lab

You are required to create a class that can detect palindromes. A palindrome is a word or sentence that spells the same forwards as backwards. For example:

- Mom
- Radar
- Race car
- Dennis sinned
- Murder for a jar of red rum
- Are we not drawn onward we few drawn onward to new era

There are many ways to detect if a phrase is a palindrome. The method that you will use in this assignment is by using one stack. This works in the following way: Push half of the characters into the stack and then compare the characters as you pop them from the stack with the second half of the word.

Note: You must ignore case when determining if the word is a palindrome or not. Thus, it does not matter at all if some of the characters are capitalised and others not. Furthermore, whitespace (spaces) also does not matter, so you must ignore any spaces. **Hint:** You may use the built-in Java methods `toLowerCase` and `split` in the `String` class: http://docs.oracle.com/javase/7/docs/api/java/lang/String.html.

Exercise 4-6 Set Datatype using Arrays

Implement a class `MySet` as described below. You are required to use arrays for storage and manipulation of data.

- The class `MySet` is used for organizing objects of type `Integer`. You should provide implementations for all the methods described below and specify the complexity of each of your implementations in Big O notation. `MySet` **is not allowed to contain duplicates**. You can assume that a `MySet` contains a maximum of 100 elements.
- `MySet` operations:
  - Construct an empty `MySet`

    ```
    public MySet()
    ```
  - Return the number of elements in `MySet`

    ```
    public int cardinality()
    ```

- Insert an `Integer` element into `MySet`

  `public void insert(int element)`

- Remove a given `int` value from `MySet`

  `public void remove(int element)`

- Check if `MySet` contains a given `int` value: return `true` if it does, return `false` otherwise.

  `public boolean contains(int element)`

- Replace a given `int x` with another given `int y`. If `x` is not in `MySet` then you should just insert `y`.

  `void replace(int x, int y)`

- Check if `MySet` is empty: return `true` if it is, return `false` otherwise.

  `public boolean isEmpty()`

Exercise 4-7 To be discussed in the Tutorial
Valid Prefix Expression - Midterm 2014

An expression in prefix notation is one where the operators are placed to the left of operands. You should decide whether a sequence corresponds to a valid prefix arithmetic expression.

For example, `* 1 2` is a valid prefix expression however `1 2 *` is not a valid expression in prefix notation.

Propose an algorithm (in Pseudocode) with linear time complexity ($O(n)$) to check the validity of an expression.