CSE 247 Data Structures (3 + 1)   Fall 2020

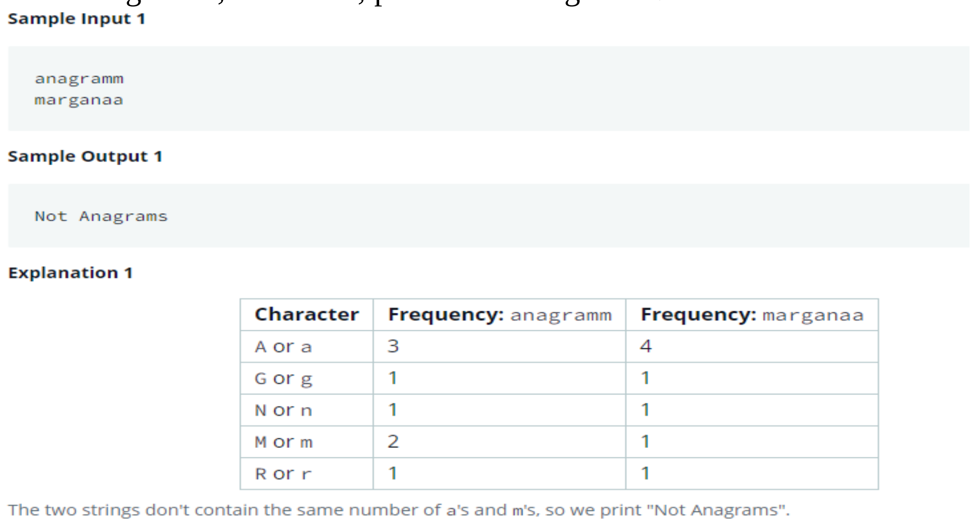
Assignment 1

Instructor: Dr. Asma Sanam Larik (asma.sanam@khi.iba.edu.pk)

Topic: Strings

Question 1: Anagrams

Two strings str1 and str2, are called anagrams if they contain all the same characters in the same frequencies. For example, the anagrams of CAT are CAT, ACT, TAC, TCA, ATC, and CTA. Write java method that would take two strings as input and if they are case-insensitive anagrams, print "Anagrams"; otherwise, print "Not Anagrams".



Sample Input 1

```
anagramm
marganaa
```

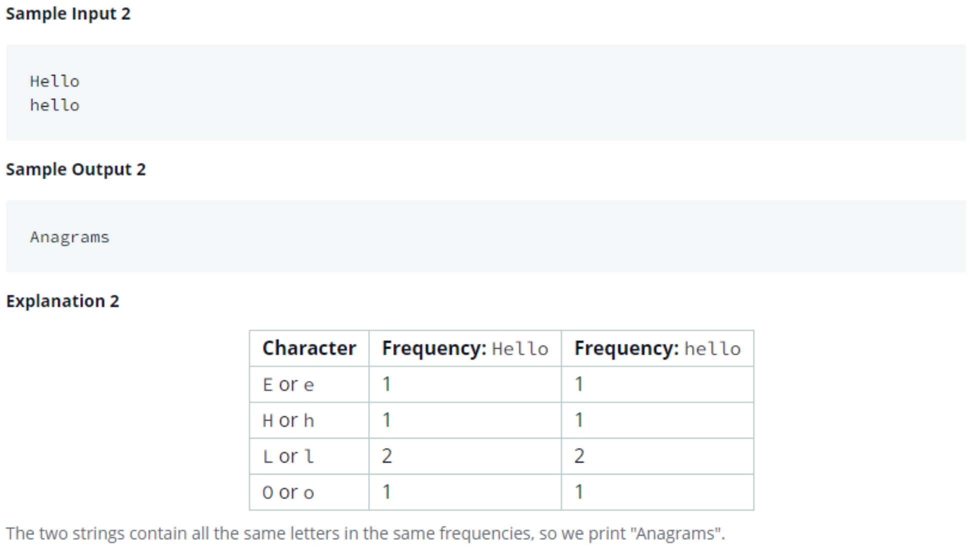
Sample Output 1

```
Not Anagrams
```

Explanation 1

Character	Frequency: anagramm	Frequency: marganaa
A or a	3	4
G or g	1	1
N or n	1	1
M or m	2	1
R or r	1	1

The two strings don't contain the same number of a's and m's, so we print "Not Anagrams".



Sample Input 2

```
Hello
hello
```

Sample Output 2

```
Anagrams
```

Explanation 2

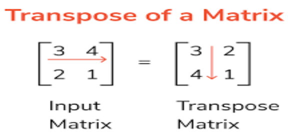
Character	Frequency: Hello	Frequency: hello
E or e	1	1
H or h	1	1
L or l	2	2
O or o	1	1

The two strings contain all the same letters in the same frequencies, so we print "Anagrams".

Topic: Arrays

Question 2: Matrix Transpose

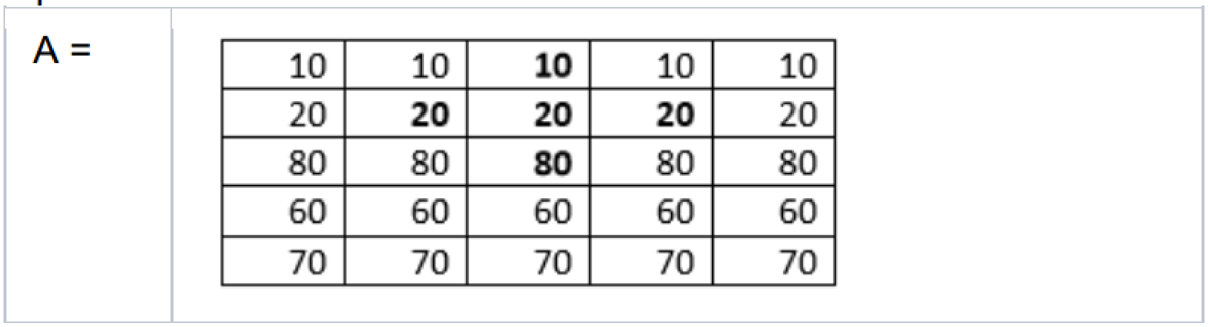
The transpose of a matrix is a new matrix whose rows are the columns of the original. Write a function matrix_transpose that takes an integer matrix A and returns the transpose of matrix A.



Question 3: Image Blurring

An image can be blurred by replacing the value of each pixel by the average of the pixel itself and its neighboring pixels. Write a function blur_image() that takes as parameter an image in the form of a 2D array and blurs it by averaging each pixel value with four of its neighboring pixels (Top, Bottom, Right, and Left).

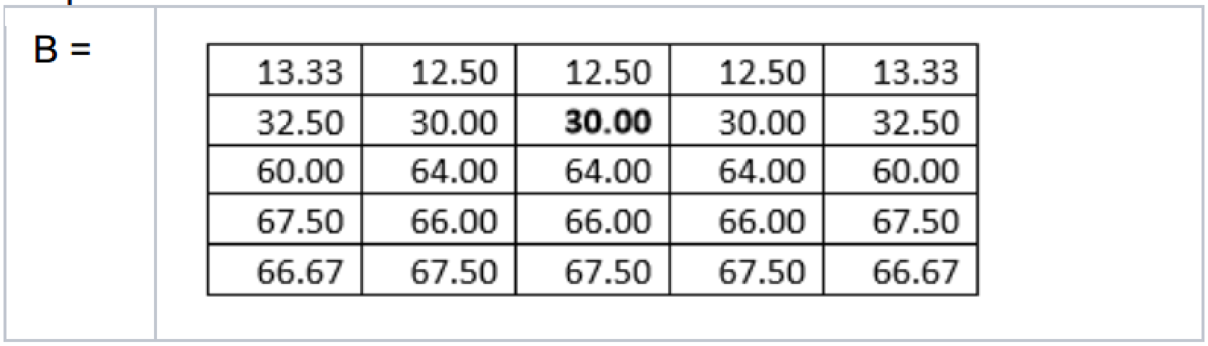
Input:



A =

10	10	**10**	10	10
20	**20**	**20**	**20**	20
80	80	**80**	80	80
60	60	60	60	60
70	70	70	70	70

Output:



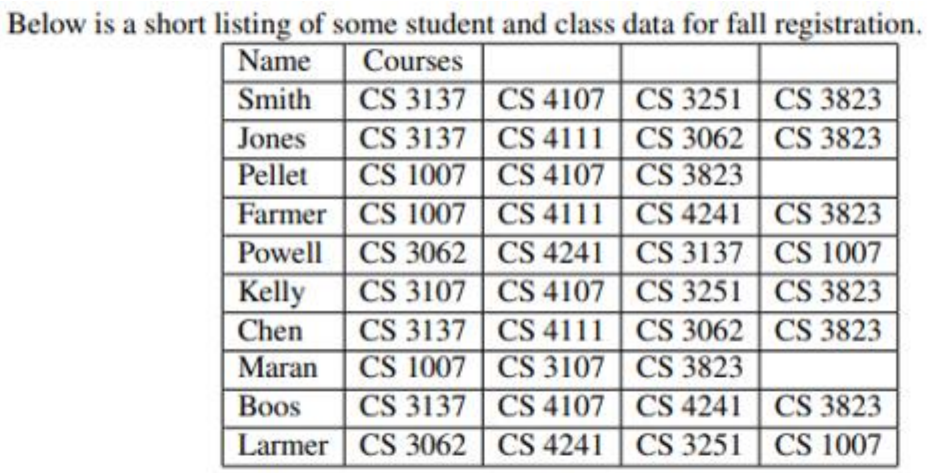
B =

13.33	12.50	12.50	12.50	13.33
32.50	30.00	**30.00**	30.00	32.50
60.00	64.00	64.00	64.00	60.00
67.50	66.00	66.00	66.00	67.50
66.67	67.50	67.50	67.50	66.67

For example: Input: A = Output: B = Where each $Bij = Aij + Aij+1 + Aij-1 + Ai+1j + Ai-1j$
For example, $B12$ = (20 + 20 + 20 + 80 + 10)/5 = 30.00
Note : Not all of the neighbors are available in boundary cases. You have to write suitable conditions accordingly.

Topic: Lists - Singly. Doubly, Circular

Question 4: Student Registration

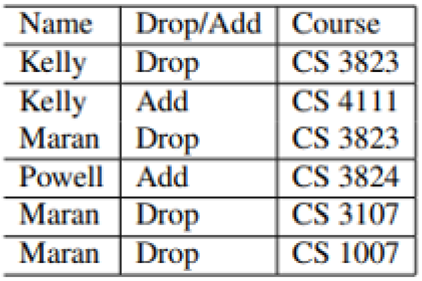


Below is a short listing of some student and class data for fall registration.

Name	Courses			
Smith	CS 3137	CS 4107	CS 3251	CS 3823
Jones	CS 3137	CS 4111	CS 3062	CS 3823
Pellet	CS 1007	CS 4107	CS 3823	
Farmer	CS 1007	CS 4111	CS 4241	CS 3823
Powell	CS 3062	CS 4241	CS 3137	CS 1007
Kelly	CS 3107	CS 4107	CS 3251	CS 3823
Chen	CS 3137	CS 4111	CS 3062	CS 3823
Maran	CS 1007	CS 3107	CS 3823	
Boos	CS 3137	CS 4107	CS 4241	CS 3823
Larmer	CS 3062	CS 4241	CS 3251	CS 1007

Describe the data structures that you would create so that the administrator could print out a list of each student registered in the course.

Using the data structure describe how a Drop /Add program would work. Input to Add/Drop would be as follows:



Name	Drop/Add	Course
Kelly	Drop	CS 3823
Kelly	Add	CS 4111
Maran	Drop	CS 3823
Powell	Add	CS 3824
Maran	Drop	CS 3107
Maran	Drop	CS 1007

Question 5: Linked List of Words

Write a method **createArrayOfLL** that takes a file of words, one in each line and creates an array of linked list nodes as follows.

1. Create an array of 26 linked lists
2. Open the file and Insert each word into the array based on its first letter, eg: any word that begins with ‘a’ goes to A[0] list etc..
3. Return the reference to the array of linked lists

Method definition is as follows:
public Node[] createArrayOfLL(String inputfile) {}

Question 6: Split of Linked list

Given a list of integer, split it into two sublists — one for the front half, and one for the back half. If the number of elements is odd, the extra element should go in the front list. So FrontBackSplit() on the list {2, 3, 5, 7, 11} should yield the two lists {2, 3, 5} and {7, 11}. Getting this right for all the cases is harder than it looks. You should check your solution against a few cases (length = 2, length = 3, length=4) to make sure that the list gets split correctly near the short-list boundary conditions. If it works right for length=4, it probably works right for length=1000. You will probably need special case codes

Topic: Algorithm Analysis

Question 7: Unimodal Sequence

Given **n ≥ 1**, a sequence of n integers **a[0],. . ., a[n−1]** is unimodal if there exists **t** (with **0 ≤ t < n**) that satisfies the following conditions:

- **a[0] ≤ . . . ≤ a[t−1] ≤ a[t]**
- **a[t] > a[t+1] > . . . > a[n−1]**

The element at is called the top of the sequence.

For example, the sequence 1, 3, 5, 9, 4, 1 is unimodal, and its top is 9 (take t = 3).

Write java code that takes a unimodal sequence ,returns the index of the top of the sequence.The solution must have cost **Θ(log(n))** in time in the worst case.

Topic: Stack

Question 8: Expression Evaluation

Assume we have a mathematical expression in postfix notation available as a string of operands and operators. The procedure for evaluating such a postfix expression is as follows:

1. Scan the expression left to right.

2. Skip values or variables (operands).

3. When an operator is found, apply the operation to the preceding two operands.

4. Replace the two operands and operator with the calculated value (three symbols are replaced with one operand).

5. Continue scanning until only a value remains -- the result of the expression

Write a java method that takes input a well-formed mathematical expression as a string in postfix notation and evaluates it using a stack. We assume the input to be already in post-fix form. Also, we are assuming only four binary operators {+, - , /, *} are allowed.

Question 9: Infix to Prefix

Infix, Postfix and Prefix notations are three different but equivalent ways of writing expressions. In Infix notations, operators are written in-between their operands. However, in Prefix expressions, the operator comes before the operands. Assume the infix expression is a string of tokens delimited by spaces. The operator tokens are *, /, +, and −, along with the left and right parentheses, (and). The operand tokens are the single-character identifiers A, B, C, and so on. The following steps will produce a string of tokens in prefix order:

1. Reverse the infix expression i.e. A + B * C will become C * B + A. Note while reversing each '(' will become ')' and each ')' becomes '('.

2. Obtain the postfix expression of the modified expression i.e. C B * A +.

3. Reverse the postfix expression. Hence in our example prefix is + A * B C.

Infix to Prefix Conversion Simulator:
https://raj457036.github.io/SimpleTools/prefixAndPostfixConvertor.html

Write a method Infix_to_Prefix that takes an arithmetic expression in Infix notation as a parameter and returns the corresponding arithmetic expression with Prefix notation.

Question 10: Prefix Expression Evaluation

Assume the prefix expression is a string of tokens delimited by spaces. The operators are *, /, +, and - and the operands are assumed to be single-digit integer values. The output will be an integer result.

1. Create an empty stack called operandStack.

2. Convert the string to a list by using the string method split.

3. Reverse the prefix expression.

4. Scan the token list from left to right.

- If the token is an operand, convert it from a string to an integer and push the value onto the operandStack.

- If the token is an operator, *, /, +, or -, it will need two operands. Pop the operandStack twice. The first pop is the first operand and the second pop is the second operand. Perform the arithmetic operation. Push the result back on the operandStack.

5. When the input expression has been completely processed, the result is on the stack. Pop the operandStack and return the value.

Prefix Expression Evaluation Simulator:
https://raj457036.github.io/SimpleTools/prefixAndPostfixEvaluator.html

Write a method EvalutePrefix that take, as input, an expression in prefix notation, and return, as output, the computed value of the given expression.

Instructions to Students:

- The assignment would be due on Sunday 18th Oct before 11:00 – No late submissions
- You are required to work individually on the assignment
- Plagiarism means zero you can take help from me, TAs, online resources

-- Good Luck --