2024 NORTHERN INDIANA HIGH SCHOOL CODING COMPETITION

Round Two

Acknowledgement: Some problems used in this competition are modified from **AP Computer Science A Exam Questions,** which are available at https://apcentral.collegeboard.org/courses/ap-computer-science-a/exam/past-exam-questions

COMPUTER SCIENCE AND INFORMATICS
COLLEGE OF LIBERAL ARTS AND SCIENCES
INDIANA UNIVERSITY SOUTH BEND

Problem 1. Sort Dates

You are given an array of Strings. Each cell contains a String representing a date in the format of **dd Mmm yyyy,** where **dd** are two digits representing the date, and **Mmm** are the first three letters of a month (the first letter is capital), and **yyyy** are 4 digits representing a year in the range of 1000 to 9999. These three fields are separated by a white space. Examples of the strings are 01 Jan 2001, 09 Feb 2099, 13 Nov 2023, 27 Apr 1999, 02 Dec 1877.

Write a program that can sort the array in increasing order of the dates. Input to this program has multiple lines. The first line is the array size (greater than or equal to 1). The rest of the lines contains the dates, each in a line. Output of the program should display all dates in increasing order with each date in one line.

Sample input (red color) and output (blue color):

Input:

```
3
01 May 2022
02 May 2021
03 Jun 2020
```

Output:

```
03 Jun 2020
02 May 2021
01 May 2022
```

Input:

```
4
01 May 2022
02 Dec 2021
03 Nov 2020
01 Oct 2020
```

Output:

```
01 Oct 2020
03 Nov 2020
02 Dec 2021
01 May 2022
```

Problem 2. Word Scramble

Write a program that accepts an English word of at most 8 letters (all in lower cases) as input and generates a list of unique scrambled words with all the letters in the original word but in different orders. The original word should be included in the output.

The output should have no duplicate words and the words should be displayed in increasing order alphabetically with one word on each line.

Sample input (red color) and output (blue color):

```
Input: abc
Output:
abc
acb
bac
bca
cab
cba
```

```
Input: cnn
Output:
cnn
ncn
nnc
```

```
Input: been
Output:
been
bene
bnee
eben
ebne
eebn
eenb
enbe
eneb
nbee
nebe
neeb
```

```
Input: uu
Output:
uu
```

Problem 3. New Array

You are given two integer arrays **A** and **B** of the same size, say **m** (**m >= 1**). These two arrays can be used to build an array **C** of size **n** (**1 <= n <= m**) such that the values in **C** are **non-decreasing** in the following way:

First, you must pick an index **i** (**0 =< i <= m-1**) for **A** and **B**. Then you choose either A[i] or B[i] and assign its value to C[0]. That is C[0] = A[i] or B[i]. Next, you choose either **A**[i+1] or **B**[i+1] and assign its value to **C**[1]. That is **C**[1] = **A**[i+1] or **B**[i+1]. Next, you choose either **A**[i+2] or **B**[i+2] and assign its value to **C**[2], so on and so forth. Finally, you choose either **A**[i+n-1] or **B**[i+n-1] and assign its value to **C**[n-1].

The procedure stops either when you reach the end of the two arrays, or when you can no longer choose an element larger than or equal to the previous one, keeping in mind that values array **C** are **non-decreasing.**

The question is, given the procedure described above, what is the longest array C that can be built out of the two given arrays A and B?

Example 1:

```
A = [2,7,3]
B = [4,2,6]
C = [2,2,3]
```

Example 2 (i starts from 0, non-optimal):

```
A = [5,2,4,1]
B = [3,6,2,5]
C = [5,6]
```

Example 3 (i starts from 1, optimal):

```
A = [5,2,4,1]
B = [3,6,2,5]
C =   [2,2,5]
```

Input to your program contains three lines. The first line contains the array size. The second and the third lines contain the elements of array A and array B (the elements are separated by spaces). The output of your program is the **maximum size** of array **C** that you can create.

Sample input (red color) and output (blue color):

Input:

```
3
2 7 3
4 2 6
```

Output: 3 (see Example 1)

Input:

```
4
5  2  4  1
3  6  2  5
```

Output: 3 (see Example 3)

Note: Your program must be able to handle large array size (less than 50) efficiently. If in 30 seconds, your program cannot produce the correct output for a testing case on the judges' machine, you will not receive credit for that testing case.