

Course Title: Computer Science II

Duration: 90 minutes (References are not allowed)

Test's Code: 03

**Question 1:**

The Adjacency Matrix of a finite undirected graph is a matrix with rows and columns labeled by graph vertices. Given that the graph vertices are zero-based numbering, the cell (i,j) is 1 or 0 according to whether i and j are adjacent or not (i and j are adjacent if there is an arc between them).

Given an adjacency matrix, write a program to do the following requirements:

- Allocate memory for the matrix.
- Find vertices that are adjacent to all other vertices using **pointer operators**.

**Input:**

- The first line is the number of vertices **n**.
- The next **n** lines are the rows of the matrix. Cell values on each row are separated by a space.

**Output:**

- A list of vertices that are adjacent to all other vertices, separated by a space.

**Examples:**

Input	Output
4 0 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0	1 2

**Question 2:**

Write a program to insert nodes into a single linked list, the value of each node is an positive integer. Insertion stops when the program encounters a node with a "0" value (this node is not inserted into the list).

Do the following requirements:

- Find the index **i** such that the number of prime numbers before **i** is equal to the number of prime numbers after **i**. If there are many satisfying indices, choose the last one.
- Suppose there is a satisfying index. Delete all prime numbers after this index.

**Input:**

- List of elements of the linked list, separated by a new line.

**Output:**

- Print out "Empty List" if the list is empty.
- Print out the inserted list.
- Print out the index **i**. If there is no satisfying index, print out "not found".

- Print out the list after deleting prime numbers after **i**.

**Examples:**

Input	Output
1	1 9 2 3 4 6 5 10 13 8
9	5
2	1 9 2 3 4 6 10 8
3	
4	
6	
5	
10	
13	
8	
0	

**Question 3:**

Implement a Stack and its operators using a Single Linked List. Each element is an integer.

Do the following requirements:

- Push a sequence of integers into the stack. Insertion stops when encountering the "0" element.
- **Use only stack or queue operators** to reverse the order of prime numbers in the stack and keep the relative positions of the other numbers unchanged.

**Input:**

- Sequence of integers separated by a new line.

**Output:**

- Print out elements of the stack separated by a space.
- Print the result stack after reversing.

Input	Output
6	4 8 3 11 10 9 5 6
5	4 8 5 11 10 9 3 6
9	
10	
11	
3	
8	
4	
0	

**Question 4:**

Given an array of **n** positive integer numbers, find the prime number nearest to the center position of the array using recursion.

**Input:**

- The first line is **n**.
- The second line is the elements of the array separated by a space.

**Output:**

- The satisfying prime number. If there are many of them, separated by a space.

**Constraint:**

- **n** is odd.

**Examples**

Input	Output
9 2 5 6 3 1 4 6 8 11	3

**Question 5:**

Given a binary square matrix, check if this matrix contains a square submatrix with all "1" values. The size of the submatrix is at least 2x2.

**Input:**

- The first line is **n**.
- The next **n** lines are the rows of the matrix. Cell values on each row are separated by a space.

**Output:**

- If there exists a square submatrix with all "1" values, print out "yes", else print "no".

**Constraint:**

- **n**  $\geq 3$

**Examples**

Input	Output
5 1 0 0 0 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 0 0 0 0	yes