Assignment 2

Problem 1: Fibonacci Puzzle

Once upon a time in a small village, a young boy named Ravi was known for his love of numbers. One day, his grandfather came to him with a puzzle. Grandfather said, "Ravi, I need your help. There is a special sequence of numbers called the Fibonacci series, where each number is the sum of the two before it. Can you find the first n numbers in this sequence for me?"

Ravi thought for a moment and decided to write a program to solve the puzzle. Can you help Ravi write a simple C program to display the first n terms of the Fibonacci series?

Input Format:

A single integer n, representing how many terms of the Fibonacci series Ravi needs to find.

Constraints:

- $3 \leq n \leq 50$

Output Format:

Print the first n terms of the Fibonacci series in a single line, separated by spaces.

Example 1:

```
Input: 5
   Output: 0 1 1 2 3
```
Copy

Explanation:

In this example, $n = 5$, which means we need to print the first five terms of the Fibonacci series.

- The Fibonacci series starts with 0 and 1.
- Each subsequent term is the sum of the two preceding ones. So, $0 + 1 = 1$, $1 + 1 = 2$, and $1 + 2 = 3$.

- Therefore, the first five terms are `0 1 1 2 3`.

Example 2:

```
Input: 7
Output: 0 1 1 2 3 5 8
```
Copy

Explanation:

In this example, `n = 7`, which means we need to print the first seven terms of the Fibonacci series.

- The Fibonacci series starts with `0` and `1`.
- The next terms are calculated as `0 + 1 = 1, 1 + 1 = 2, 1 + 2 = 3, 2 + 3 = 5,` and `3 + 5 = 8`.
- Therefore, the first seven terms are `0 1 1 2 3 5 8`.

Problem 2: Jolly Number

Write an algorithm to determine Jolly numbers up to N. A Jolly number is defined by the following process:

1. Starting with any positive integer, replace the number by the sum of the squares of its digits.
2. Repeat the process until the number equals 1 (where it will stay), or it loops until a maximum of 100 iterations.
3. Those numbers for which this process ends in 1 are called Jolly numbers.

Print all Jolly Numbers from 1 up to N (included).

Input Format:

A single integer `N`.

Output Format:

A list of all Jolly numbers up to N separated by spaces.

Constraints:

- $1 \leq N \leq 10{,}000$

Examples:

```
Input:
19
Output:
1 7 10 13 19
```
Copy

Note:

The number 19 undergoes the following transformations until it reaches the Jolly condition (sum of the squares of its digits results in 1):

1. $1^2 + 9^2 = 82$
2. $8^2 + 2^2 = 68$
3. $6^2 + 8^2 = 100$
4. $1^2 + 0^2 + 0^2 = 1$

This sequence shows that starting from 19, the sequence reaches 1 in four iterations, thus confirming 19 as a Jolly number.

```
Input:
2
Output:
1
```
Copy

Problem 3: Devesh Numbers

A Devesh number is a positive integer that is divisible by the sum of its digits. For example, 12 is a Devesh number because the sum of its digits is 1 + 2 = 3, and 12 is divisible by 3. Similarly, 18 is a Devesh number because the sum of its digits is 1 + 8 = 9, and 18 is divisible by 9.

Task

Write a program that takes an integer N as input and outputs all the Devesh numbers less

than or equal to N.

Input Format:

- Two integers A and B where
- 1 ≤ A ≤ 100,000
- 1 ≤ B ≤ 100,000
- A ≤ B

Output Format:

- A list of all Devesh numbers within the range A to B (both included) separated by spaces.

Examples:

```
Input: 1 30                                                    Copy
Output: 1 2 3 4 5 6 7 8 9 10 12 18 20 21 24 27 30
```

Explanation:

The Devesh numbers less than or equal to 30 are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 18, 20, 21, 24, 27, 30.

Problem 4: Sudoku Solver

Determine if a 9 × 9 Sudoku board is valid according to the following rules:

1. Each row must contain the digits 1-9 without repetition.
2. Each column must contain the digits 1-9 without repetition.

Note: You only need to check the above-mentioned rules to solve the problem.

Task

Write a program that takes a 9x9 2D array as input and determines whether it is a valid Sudoku solution.

Input Format:

- A 2D array with dimensions 9x9 containing digits from range 1 to 9.

Output Format:

- Output `1` if the Sudoku is valid, otherwise `0`.

Examples:

Input:

```
6 5 9 3 1 4 2 8 7
1 8 7 6 5 2 4 3 9
2 3 4 8 9 7 5 1 6
4 2 6 1 3 5 9 7 8
8 7 1 9 4 6 3 5 2
5 9 3 2 7 8 6 4 1
3 1 2 5 8 9 7 6 4
7 6 5 4 2 1 8 9 3
9 4 8 7 6 3 1 2 5
```

Output: `1`

Explanation: The matrix follows the rules of Sudoku.

Input:

```
5 3 4 6 5 8 9 1 2
6 7 2 1 9 5 3 4 8
1 9 8 3 4 2 5 6 7
8 5 9 7 6 1 4 2 3
4 2 6 8 2 3 7 9 1
7 1 3 9 2 4 8 5 6
9 6 1 5 3 7 2 8 4
2 8 7 4 8 9 6 3 5
3 4 4 2 8 6 1 7 9
```

Output: `0`

Explanation: The first row has two occurrences of 5. The fifth column has two occurrences of 8.

Submission Guidelines

- Do not rename any files given in the handout. Only write the code in the specified C files in the respective directories.