# Problem Solving Sessions

- [x] Day 1: Problem Solving Techniques
- [x] Day 2: Patterns
- [x] Day 3: Arrays - Basics [ ] Day 4: Arrays - Advanced + Time and Space Complexity [ ] Day 5: Strings - Basics
- [ ] Day 6: Strings - Advanced
- [ ] Day 7: Sorting Algorithms
- [ ] Day 8: Searching Algorithms
- [ ] Day 9: Linked Lists
- [ ] Day 10: Stacks and Queues

## Day 1: Problem Solving Techniques + Time and Space Complexity

- Problem Solving Techniques
- Interview Problem - How to approach and solve
- Time Complexity - Big O Notation

**Question:** Given a number, Check if the number is a even number or an odd number.

Approach 1: Divisibility by 2

1. Take a number as input <- number
2. Find the remainder of the number when divided by 2
3. If the remainder is 0, then the number is even, else the number is odd

Algorithm 2: Multiplying by 5

1. Take a number as input <- number
2. Multiply the number by 5
3. If the last digit of the number is 0 or 5, then the number is even, else the number is odd

Algorithm 3: Bitwise AND

1. Take a number as input <- number
2. Perform a bitwise AND operation between the number and 1
3. Step 2 will give 1 if the last bit of the number is 1, else 0
4. If the result is 1, then the number is odd, else the number is even

Algorithm 4: Checking the last digit

1. Take a number as input <- number
2. Check the last digit of the number
3. If the last digit is 0, 2, 4, 6, or 8, then the number is even, else the number is odd

Problem: Given two numbers, check whether they are amicable numbers or not.

Example:

220 and 284 are amicable numbers.

Divisors of 220: 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110 Sum of divisors of 220: 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284

Divisors of 284: 1, 2, 4, 71, 142 Sum of divisors of 284: 1 + 2 + 4 + 71 + 142 = 220

In summary, the sum of the divisors of the first number is equal to the second number and the sum of the divisors of the second number is equal to the first number.

Algorithm: Brute Force Technique

1. Take two numbers as input <- number1, number2
2. Find the divisors of the first number <- divisors1
3. Find the divisors of the second number <- divisors2
4. Find the sum of the divisors of the first number <- sum1
5. Find the sum of the divisors of the second number <- sum2
6. If sum1 is equal to number2 and sum2 is equal to number1, then the numbers are amicable numbers
7. Else, the numbers are not amicable numbers

Testing the algorithm with multiple test cases:

Positive Test Cases: These are the test cases where the algorithm should work as expected.

Input:

220 284

Output:

Amicable numbers

Input:

1184 1210

Output:

Amicable numbers

Negative Test Cases: These are the test cases where the algorithm should not work as expected.

Input:

220 285

Output:

Not amicable numbers

Input:

1184 1211

Output:

Not amicable numbers

Boundary Test Cases: These are the test cases where the algorithm should work at the edge cases.

Input:

1 1

Output:

Not amicable numbers

Input:

1 2

Output:

Not amicable numbers

Edge Test Cases: These are the test cases where the algorithm should work at the extreme edge cases.

Input:

0 0

Output:

Not amicable numbers

Input:

0 1

Output:

Not amicable numbers

## Problem Solving Techniques

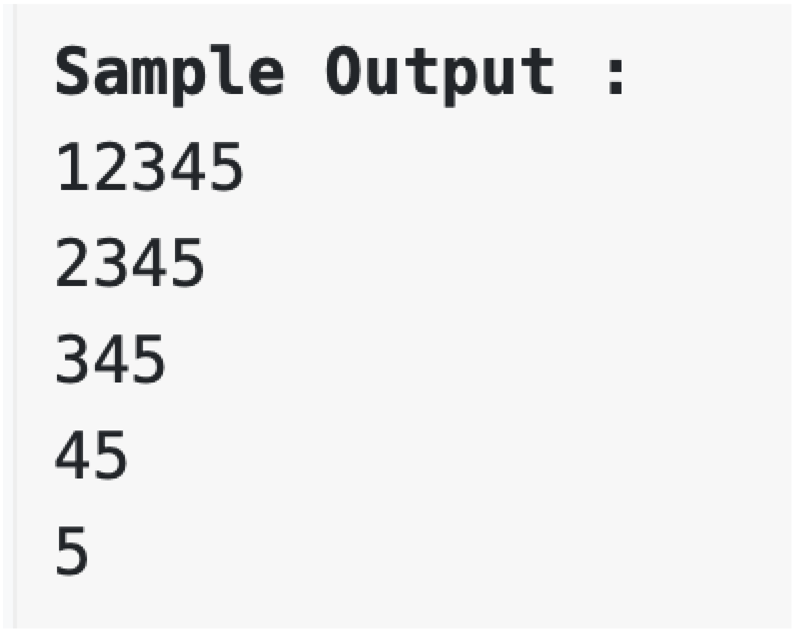
1. Understand the problem statement
2. Dry run the problem with an example
3. Write the algorithm
4. Testing the algorithm with multiple test cases
5. Write the code for the algorithm
6. Test the code with multiple test cases
7. Analyze the time and space complexity of the code
8. Optimize the code if possible

> Note: In above steps, whenever possible, we need to break the problem into smaller subproblems and solve them individually

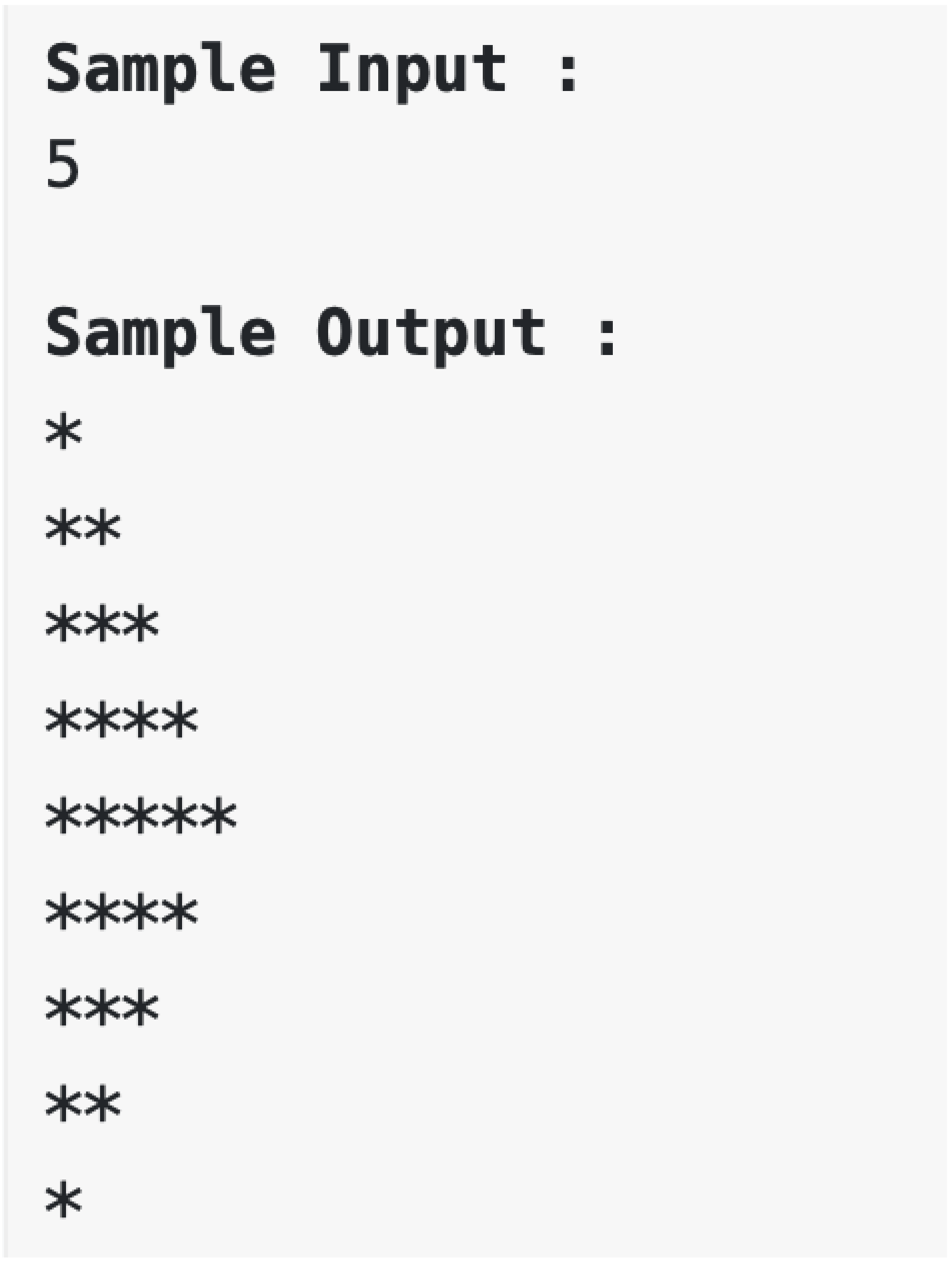
## Homeworks

1. Given a number, check if it is a prime number or not.
2. Given a number, check if it is a palindrome number or not.
3. Given a number, check if it is a perfect number or not.
4. Given a number, check if it is a happy number or not.
5. Given a number, check if it is a fibonacci number or not.
6. Given a number, check if it is a armstrong number or not.
7. Given two numbers, find the greatest common divisor (GCD) of the two numbers.
8. Given two numbers, find the least common multiple (LCM) of the two numbers.
9. Given a number, find the factorial of the number.
10. Given a number, find the sum of the digits of the number.
11. Given a number, find the reverse of the number.
12. Given a number, find the number of digits in the number.
13. Given a number, find the number of prime factors of the number.
14. Find the nth number in the fibonacci sequence.
15. Find the nth prime number.

### Day 2: Patterns

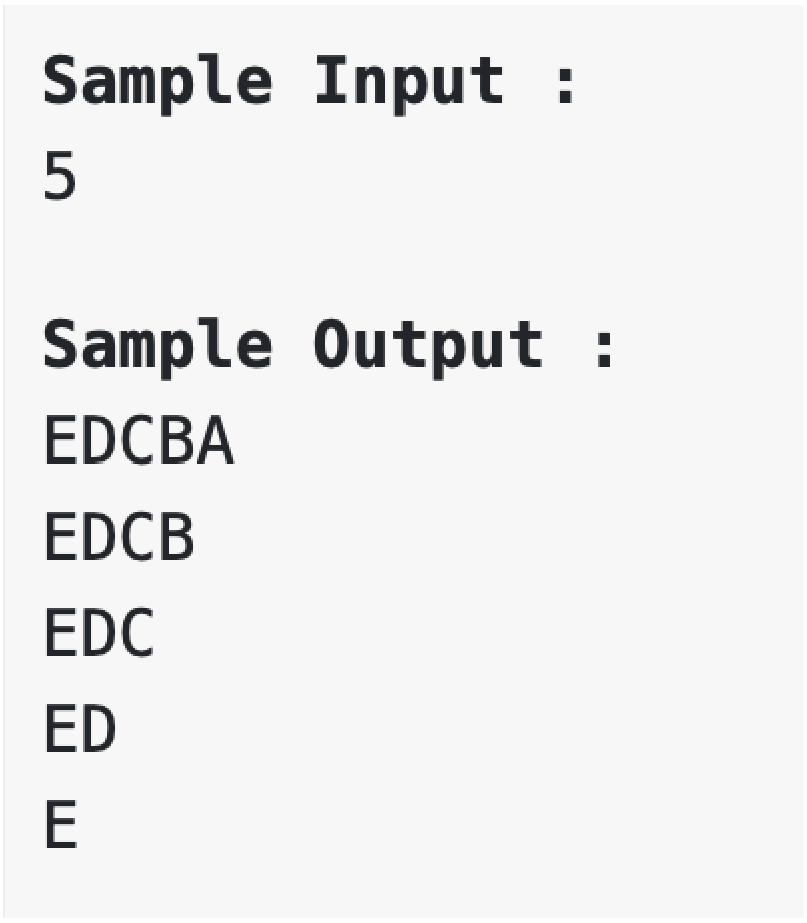


```
Sample Output :
12345
2345
345
45
5
```



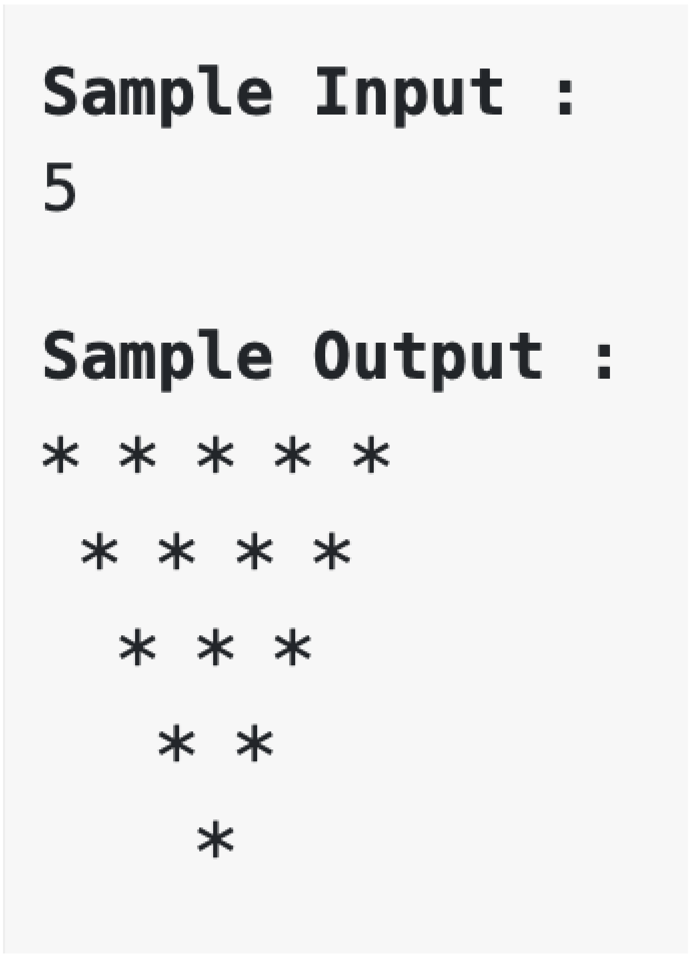
```
Sample Input :
5

Sample Output :
*
**
***
****
*****
****
***
**
*
```



```
Sample Input :
5

Sample Output :
EDCBA
EDCB
EDC
ED
E
```

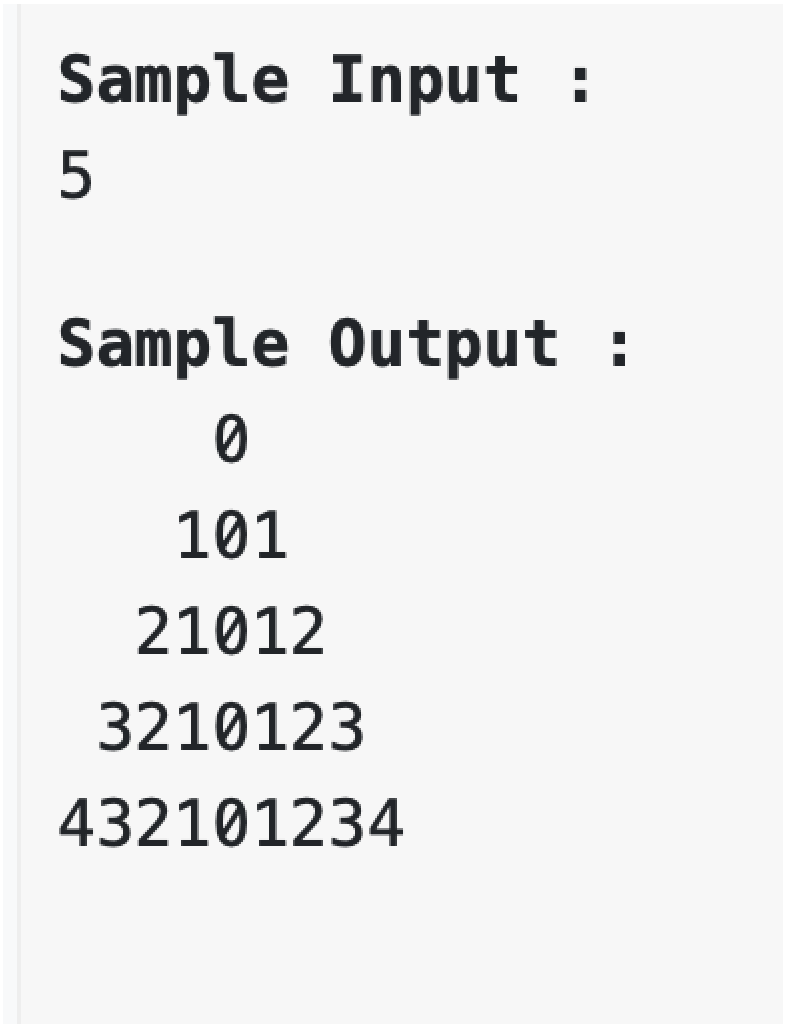


```
Sample Input :
5

Sample Output :
* * * * *
 * * * *
  * * *
   * *
    *
```

```
Sample Input :
5

Sample Output :
    1
   1 2
  1   3
 1     4
1 2 3 4 5
```



```
Sample Input :
5

Sample Output :
    0
   101
  21012
 3210123
432101234
```

## Day 3: Arrays - Basics

### Homeworks

1. Given an array of integers, find the frequency of each element in the array.
2. Given an array of integers, find the number of duplicates in the array.
3. Given an array of integers, find the number of unique elements in the array.
4. Given an array of integers, remove the duplicates from the array.
5. Given an array of integers, find the first non-repeating element in the array.
6. Given an array of integers, find the first repeating element in the array.
7. Given an array of integers, find the second largest element in the array.
8. Given an array of integers, find the maximum frequency of an element in the array.
9. Given an array of integers, find the minimum frequency of an element in the array.
10. Given a String, find the frequency of each character in the String and print as like 'a2b3c4d5'.
11. Given a string, find the first non-repeating character in the string.
12. Given a string, find the first repeating character in the string.
13. Given two arrays, find the common elements between the two arrays.
14. Given two arrays, find the union of the two arrays.
15. Given two arrays, find the intersection of the two arrays.
16. Given an array of integers and the number, find only the factors of the number in the array.
17. Given two sorted arrays, merge the two arrays into a single sorted arrays.