7 . Area of Triangle

Accept the lengths of the three sides of a triangle. Calculate the area using Heron's formula and print it.

Example 1:

Input: `s1 = 5, s2 = 7, s3 = 9`

Output: `Area: 17.412 sq units`

8 . Greatest between two

Accept two numbers and determine which one is greater. Print the greatest number.

Example 1:

Input: `a = 9, b = 12`

Output: `12`

Example 2:

Input: `a = 100, b = 67`

Output: `100`

9 . Even or Odd

Accept an integer and determine whether it is an even number or odd. Print the result.

Example 1:

Input: a = 9

Output: 9 is an odd number

Example 2:

Input: 24

Output: 24 is a odd number

12 . Leap Year

Accept a year from the user and determine if it is a leap year or not based on the leap year conditions.

If the year is evenly divisible by 4, it is a leap year.
If the year is evenly divisible by 100, it is not a leap year, unless:
The year is also evenly divisible by 400, then it is a leap year.

Example 1 :

```
Input: year = 2024
Output: 2024 is a leap year
```

Example 2 :

```
Input: year = 1800
Output: 2021 is not a leap year
```

13 . Electricity Bill

A utility company needs a program to calculate electricity bills. Accept the number of units consumed from the user and calculate the bill amount based on the following criteria:

For the first 100 units, the rate is rupees 4.2 per unit.
For the next 100 units, the rate is rupees 6 per unit.
For the next 200 units, the rate is rupees 8 per unit.
For units above 400, the rate is rupees 13 per unit.

Example 1 :

Input: 150

Output: Total bill amount: 720

Explanation: 100 * 4.2 + 50 * 6 = 720

Example 2 :

Input: 700

Output: Total bill amount: 6520

14. Shop Discount

You are tasked with creating a shop discount program that calculates discounts based on the total price of items purchased by a customer.

0 >= totalPrice <= 5000 - 0% discount
5000 > totalPrice <= 7000 - 5% discount
7000 > totalPrice <= 9000 - 10% discount
9000>totalPrice - 20% discount

Example 1:

Input: `8000`

Output: `Payable amount is 7200`

Example 2:

Input: `11000`

Output: `Payable amount is 8800`

15 . N times “hello world”

Accept an integer and Print hello world n times.

Example 1:

```
Input: n = 5

Output: hello world
        hello world
        hello world
        hello world
        hello world
```

16 . N natural numbers

Write a program to print all natural numbers up to a given limit 'n'.

Example 1:

Input: n = 5

Output: 1 2 3 4 5

Example 2:

Input: n = 10

Output: 1 2 3 4 5 6 7 8 9 10

17 . Sum of first N numbers

Write a program to calculate the sum of the first 'n' natural numbers.

Example 1:

Input: n = 5

Output: Sum of the first 5 natural numbers: 15

Example 2:

Input: n = 10

Output: Sum of the first 5 natural numbers: 55

18. Factorial of a number

Write a program to calculate the factorial of a given number.

Example 1:

Input: n = 5

Output: 120

Example 2:

Input: n = 4

Output: 24

19 . Prime Number

Write a program to check if a given number is a prime number or not, considering the following conditions:
A prime number is a positive integer greater than 1, which has only two positive divisors: 1 and number itself.

Example 1 :

Input: `n = 17`

Output: `17 is a prime number`

Example 2:

Input: `n = 18`

Output: `18 is not a prime number`

20 . Sum of digit

Write a program to calculate the sum of the digits of a given number.

Example 1:

Input: n = 1234

Output: 1 + 2 + 3 + 4 = 10

Example 2:

Input: n = 897

Output: 8 + 9 + 7 = 24

21. Reverse the number

Write a program to accept a number and print its reverse. Input will not contain trailing zero.

Example 1:

Input: n = 12345

Output: 54321

Example 2:

Input: n = 876

Output: 678

22 . Palindrome number

Write a program to accept a number and determine if it is a palindromic number. A palindromic number is a number that remains the same when its digits are reversed.

Example 1:

Input: n = 12321

Output: 12321 is a palindromic number

Example 2:

Input: n = 123

Output: 123 is not a palindromic number

23 . Strong number

Write a program to accept a number and determine if it is a strong number or not. A strong number is a number whose sum of the factorial of each digit equals the number itself.

Example 1:

Input: n = 145

Output: 145 is a strong number

Explanation:

Sum of factorials = 1! + 4! + 5! = 1 + 24 + 120 = 145
Since the sum of factorials equals the number itself, 145 is a strong number.

Example 2:

Input: n = 123

Output: 123 is not a strong number

24 . Automorphic Number

Write a program to determine if a given number is an automorphic number. An automorphic number is a number whose square ends with the same digits as the number itself.

Example 1:

Input: n = 25

Output: 25 is a automorphic number

Explanation:

The square of 25 is 625.
Since the last two digits of the square match the original number, 25 is an automorphic number.

Example 2:

Input: n = 625

Output: 625 is a automorphic number

25 . Single Digit

Sum of Digits until the number becomes single digit.

Example 1:

Input: 6758

Output: 8

Explanation:

```
6 + 7 + 5 + 8 = 26
2 + 6 = 8
```

26 . Fibonacci series

Write a program to generate the Fibonacci series up to a given limit 'n'. The Fibonacci series is a sequence of numbers where each number is the sum of the two preceding ones, starting from 0 and 1.

Example 1 :

Input: `n = 8`

Output: `0 1 1 2 3 5 8 13`

Explanation:

```
The Fibonacci series starts with 0 and 1.
Each subsequent number in the series is the sum of the
two preceding numbers.
For example, the first few numbers in the Fibonacci
series are 0, 1, 1, 2, 3, 5, 8, 13, ...
```

Example 2 :

Input: `n = 10`

Output: `0 1 1 2 3 5 8 13 21 34`

27 . Repeat Hello

Write a program that repeatedly prints "Hello" until the user provides incorrect input using a do-while loop.

Example 1:

```
Flow:   Hello
        Press 1 to repeat
        1
        Hello
        Press 1 to repeat
        0
        Program Exits
```

28 . Calculator

Write a program to create a choice-based calculator using a do-while loop. The program should repeatedly prompt the user to choose an operation and perform the corresponding calculation until the user chooses to exit.

Example 1:

Flow:

```
Choose an operation:
Addition
Subtraction
Multiplication
Division
Exit
Enter your choice: 1 (user input)

Enter the first number: 10
Enter the second number: 5

Sum = 15 (output)
```

29 . Guess the number

Write a program that generates a random number “target” in the range of 1-100 and asks the user to guess what the number is.
If user's guess is equal to target, print “Congrats”.
if user’s guess is higher than target, print "Too high, try again."
else print "Too low, try again."
The program should use a loop that repeats until the user correctly guesses the random number.

Example 1:

```
Flow:   Guess the number
        23 (user input)
        Too low, try again
        67 (user input)
        Too high, try again
        40 (user input)
        Congrats.
        Program exits.
```

30 . Pattern - Right Triangle (Star)

Accept an integer n and print the following pattern for n.

Example 1 :

```
Input: n = 5
Output:  *
         * *
         * * *
         * * * *
         * * * * *
```

Example 2 :

```
Input: n = 4
Output:  *
         * *
         * * *
         * * * *
```

31 . Pattern - Right Triangle (Number)

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

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

32 . Pattern - Right Triangle (Alphabet)

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
A
A B
A B C
A B C D
A B C D E
```

33 . Inverted Right Triangle

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
* * * * *
* * * *
* * *
* *
*
```

34 . Mirrored Right Triangle

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
        *
      * *
    * * *
  * * * *
* * * * *
```

35 . Triangle

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
        *
      * *
    * * *
  * * * *
* * * * *
```

36 . Pattern V

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
*       *
 *     *
  *   *
   * *
    *
```

Constraints:

- n >= 3
- n will be an odd integer

36 . Pattern X

Accept an integer n and print the following pattern for n.

Example 1:

Input: n = 5

Output:

```
*       *
  *   *
    *
  *   *
*       *
```

Constraints:

- n >= 3
- n will be an odd integer

37 . Sum of array elements

Write a program that accepts the size of an array 'n' from the user, creates an array nums of size 'n', takes 'n' inputs into the array, and then prints the sum and mean of all elements.

Example 1 :

Input: `n = 5, nums = [10, 15, 20, 25, 30]`

Output: `Sum = 100, Mean = 20.0`

Example 2 :

Input: `n = 2, nums = [1, 8]`

Output: `Sum = 9, Mean = 4.5`

Constraints:

- `1 <= n <= 1000`
- `1 <= nums[i] <= 1000`

38 . Greatest in array

Write a program to find the greatest element in a given array and print its index.

Example 1 :

Input: `n = 5, nums = [10, 5, 2, 37, 8]`

Output: `Greatest number is 37 at index 3`

Example 2 :

Input: `n = 4, nums = [1, 8, 2, 6]`

Output: `Greatest number is 8 at index 1`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

39 . Second greatest in array

Create a program to find the second greatest element in a given array.

Example 1 :

Input: `n = 5, nums = [10, 5, 2, 37, 8]`

Output: `Second greatest number is 10`

Example 2 :

Input: `n = 4, nums = [1, 8, 2, 6]`

Output: `Second greatest number is 6`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`
- `nums[i] is unique`

40 . Is array sorted

Write a program to check if an array is sorted in increasing order or not.

Example 1 :

Input: `n = 5, nums = [1, 2, 3, 4, 5]`

Output: `true`

Example 2 :

Input: `n = 4, nums = [5, 2, 8, 4]`

Output: `false`

Example 3 :

Input: `n = 6, nums = [2, 3, 3, 6, 8, 8]`

Output: `true`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

41. Array deep copy

Accept an array nums of size n and make a duplicate array of nums, temp. Print all the elements of temp.

Example 1:

Input: `n = 5, nums = [1, 2, 3, 4, 5]`

Output: `[1, 2, 3, 4, 5]`

Example 2:

Input: `n = 4, nums = [5, 2, 8, 4]`

Output: `[5, 2, 8, 4]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

42 . Left rotate array

Write a program to perform a left rotation on an array by one position.

Example 1 :

Input: `n = 5, nums = [1, 2, 3, 4, 5]`

Output: `[2, 3, 4, 5, 1]`

Example 2 :

Input: `n = 4, nums = [5, 2, 8, 4]`

Output: `[2, 8, 4, 5]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

43 . Left rotate array by k

Write a program to perform a left rotation on an array by a specified number of positions k.

Example 1 :

Input: `n = 5, nums = [1, 2, 3, 4, 5], k = 2`

Output: `[3, 4, 5, 1, 2]`

Example 2 :

Input: `n = 3, nums = [2, 8, 4], k = 4`

Output: `[8, 4, 2]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`
- `0 <= k <= 1000`

44 . Reverse array

Write a program to reverse an array without using any additional space.

Example 1 :

Input: `n = 5, nums = [1, 2, 3, 4, 5]`

Output: `[5, 4, 3, 2, 1]`

Example 2 :

Input: `n = 3, nums = [2, 8, 4]`

Output: `[4, 8, 2]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`
- `reversal operation should be performed in-place`

45 . Linear Search

Write a program to accept an array and a key. Print index if key is found in the array else print -1.

Example 1 :

Input: `nums = [1, 2, 3, 4, 5], key = 4`

Output: `3`

Example 2 :

Input: `nums = [2, 8, 4], key = 9`

Output: `-1`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

46 . Binary Search

Write a program to accept an array and a key. Print index if key is found in the array else print -1 using Binary Search Algorithm.

Example 1 :

Input: `nums = [1, 2, 3, 4, 5], key = 4`

Output: `3`

Example 2 :

Input: `nums = [2, 8, 4], key = 9`

Output: `-1`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`
- `array is sorted in increasing order`

47 . Bubble Sort

Write a program to accept an array. Sort the elements of array in ascending order using Bubble sort algorithm.

Example 1 :

Input: `nums = [4, 3, 6, 7, 11, 9]`

Output: `[3, 4, 6, 7, 9, 11]`

Example 2 :

Input: `nums = [2, 8, 2, -8]`

Output: `[-8, 2, 2, 8]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

48 . Insertion Sort

Write a program to accept an array. Sort the elements of array in ascending order using Insertion sort algorithm.

Example 1 :

Input: `nums = [4, 3, 6, 7, 11, 9]`

Output: `[3, 4, 6, 7, 9, 11]`

Example 2 :

Input: `nums = [2, 8, 2, -8]`

Output: `[-8, 2, 2, 8]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

49 . Selection Sort

Write a program to accept an array. Sort the elements of array in ascending order using Selection sort algorithm.

Example 1 :

Input: `nums = [4, 3, 6, 7, 11, 9]`

Output: `[3, 4, 6, 7, 9, 11]`

Example 2 :

Input: `nums = [2, 8, 2, -8]`

Output: `[-8, 2, 2, 8]`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

50 . Count subarrays' sum equals target

Write a program to accept an array and target. Print the count of subarrays whose elements sum is equal to the target.

Example 1 :

Input: `nums = [1, 2, 3, 7, 5], target = 12`

Output: `2`

Explanation: `Subarrays [2, 3, 7] & [7, 5] sum 12`

Example 2 :

Input: `nums = [-1, 1, -1, 1, -1], target = 0`

Output: `6`

Constraints:

- `1 <= n <= 1000`
- `-1000 <= nums[i] <= 1000`

51. Separate the characters

Accept a string from the user and print each character on a new line.

Example 1:

```
Input: str = “hello”

Output: h
        e
        l
        l
        o
```

52 . Reverse string

Accept a string and print it in reverse order.

Example 1:

Input: str = “hello”

Output: “olleh”

Example 2:

Input: str = “shery”

Output: “yrehs”

53 . Palindromic string

Check if the string is Palindromic or not. Palindromic string is one whose reverse is equal to the original string

Example 1:

Input: `str = “madam”`

Output: `true`

Example 2:

Input: `str = “shery”`

Output: `false`

54 . Count vowels & consonant

Accept a string str, count & print the number of vowels, consonants, and spaces in it.

Example 1:

Input: `str = “hello world”`

Output: `Vowels: 3, Consonants: 7, Spaces: 1`

Example 2:

Input: `str = “12345”`

Output: `Vowels: 0, Consonants: 0, Spaces: 0`

55 . Toggle characters

Accept a string and toggle each character of it and form a new string. Toggle means to flip the lower case letter to upper case and vice-versa

Example 1:

Input: str = “AcgDfD”

Output: “aCGdFd”

Example 2:

Input: str = “zzzz”

Output: “ZZZZ”

Constraints:

- str contains only english alphabets

56 . Count the valid words

Take an array of string words and a string pref. Print the number of strings in words that contain pref as a prefix.

Example 1:

Input: `words = ["pay", "attire", "practice", "attend"], pref = "at"`

Output: `2`

Explanation: `words starting with pref = attend, attire`

57 . Split and change

Accept a space-separated sentence and split it into words. Print each word on a new line with the first letter capitalised.

Example 1:

Input: sentence = “Hello bhai kaise ho”

Output:
```
Hello
Bhai
Kaise
Ho
```

58 . Count frequency

Accept a string and print the frequency of each character present in the string.

Example 1:

```
Input: str = “hello”

Output:  h: 1
         e: 1
         l: 2
         o: 1
```

59 . Check Anagrams

Given two strings s and t, print true if t is an anagram of s, and false otherwise.

An Anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Example 1 :

Input: `s = "anagram", t = "nagaram"`

Output: `true`

Example 2 :

Input: `s = "debitcard", t = "badcredit"`

Output: `true`

Example 3 :

Input: `s = "naruto", t = "tanjiro"`

Output: `false`

Constraints:

- `s and t consist of lowercase English letters.`

60 . Sort the words of sentence

Accept a sentence in which words are separated by single space. Sort the words of sentence lexicographically and print the newly formed sentence.

Example 1 :

Input: sentence = “ek machli paani me gayi chapaak”

Output: “chapaak ek gayi machli me paani”

Example 2 :

Input: sentence = "quick brown fox jumps into the river"

Output: “brown fox into jumps quick river the”