```
1. Validate Balanced Brackets
```

```
using System;
using System.Text.RegularExpressions;

class Program
{
    static void Main()
    {
        Console.Write("Enter a string with brackets: ");
        string input = Console.ReadLine();


        string pattern = @"\(\)|\[\]|\{}";


        while (Regex.IsMatch(input, pattern))
        {
            input = Regex.Replace(input, pattern, "");
        }

        if (input.Length == 0)
            Console.WriteLine("True");
        else
            Console.WriteLine("False");
    }
}
```

```
2.
```

```
using System;
using System.Linq;

class Program
{
    static void Main()
    {
        Console.Write("Enter a string: ");
        string input = Console.ReadLine();

        // Find the first character whose count in the string is 1
        char? result = input.FirstOrDefault(c => input.Count(x => x == c) == 1);

        if (result != '\0')  // '\0' means no character found
            Console.WriteLine($"{result}");
        else
            Console.WriteLine("null");
    }
}
```

3.

```
using System;
using System.Linq;

class Program
{
    static void Main()
    {

        string input1 = Console.ReadLine();
        int[] arr1 = string.IsNullOrWhiteSpace(input1)
                     ? new int[0]
                     : input1.Split().Select(int.Parse).ToArray();


        string input2 = Console.ReadLine();
        int[] arr2 = string.IsNullOrWhiteSpace(input2)
                     ? new int[0]
                     : input2.Split().Select(int.Parse).ToArray();


        int[] merged = arr1.Concat(arr2).Distinct().OrderBy(x => x).ToArray();


        Console.WriteLine(string.Join(" ", merged));
    }
}
```

4.

```
using System;
class Program
{
  static void Main()
  {
      // Read array from user
      string[] input = Console.ReadLine().Split();
      int n = input.Length;
      int[] arr = new int[n];
      for (int i = 0; i < n; i++)
```

```
          arr[i] = int.Parse(input[i]);
      // Read target sum
      int target = int.Parse(Console.ReadLine());

      int count = 0;
      // Count pairs
      for (int i = 0; i < n; i++)
      {
          for (int j = i + 1; j < n; j++)
          {
              if (arr[i] + arr[j] == target)
                  count++;
          }
      }
      Console.WriteLine(count);
}}
```

5.

```
using System;
using System.Collections.Generic;

class Program
{
  static void Main()
  {
      string[] input = Console.ReadLine().Split();
      int n = input.Length;
      int[] arr = new int[n];
      for (int i = 0; i < n; i++)
          arr[i] = int.Parse(input[i]);

      HashSet<int> set = new HashSet<int>(arr);
      int longest = 0;
      foreach (int num in arr)
      {
          if (!set.Contains(num - 1)) // start of a sequence
          {
              int current = num;
              int length = 1;
              while (set.Contains(current + 1))
              {
                  current++;
                  length++;
              }
              if (length > longest)
```

```
               longest = length;
           }
      }
      Console.WriteLine(longest);
  }
}
```

6.

```
  using System;
class Program
{
  static void Main()
  {
      string[] input = Console.ReadLine().Split();
      int n = input.Length;
      int[] arr = new int[n];
      for (int i = 0; i < n; i++)
           arr[i] = int.Parse(input[i]);
      int majority = 0;
      bool found = false;
      for (int i = 0; i < n; i++)
      {
           int count = 0;
           for (int j = 0; j < n; j++)
           {
               if (arr[i] == arr[j])
                   count++;
           }
           if (count > n / 2)
           {
               majority = arr[i];
               found = true;
               break;
           }
      }
      Console.WriteLine(found ? majority.ToString() : "null");
  }
}
```

7.

```
using System;
```

```
class Program
{
    static void Main()
    {
        string input = Console.ReadLine();
        int[] arr = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

        int n = arr.Length;
        int total = 1 << n; // 2^n subsets

        Console.Write("[");
        for (int i = 0; i < total; i++)
        {
            Console.Write("[");
            bool first = true;
            for (int j = 0; j < n; j++)
            {
                if ((i & (1 << j)) != 0)
                {
                    if (!first) Console.Write(",");
                    Console.Write(arr[j]);
                    first = false;
                }
            }
            Console.Write("]");
            if (i < total - 1) Console.Write(", ");
        }
        Console.WriteLine("]");
    }
}
```

8.

```
using System;

public class Program
{
    public static void Main()
    {
        int[] arr = Array.ConvertAll(Console.ReadLine().Split(), int.Parse);
        int target = int.Parse(Console.ReadLine());

        int left = 0, right = arr.Length - 1;
        while (left <= right)
        {
            int mid = (left + right) / 2;
            if (arr[mid] == target) { Console.WriteLine(mid); return; }

            if (arr[left] <= arr[mid])
            {
                if (target >= arr[left] && target < arr[mid]) right = mid - 1;
                else left = mid + 1;
            }
            else
            {
                if (target > arr[mid] && target <= arr[right]) left = mid + 1;
                else right = mid - 1;
            }
        }

        Console.WriteLine(-1);
    }
}
```

9.

```
using System;
using System.Collections.Generic;

public class Program
{
    public static void Main()
    {
        int[] arr = Array.ConvertAll(Console.ReadLine().Split(), int.Parse);

        // Count frequency
        Dictionary<int, int> freq = new Dictionary<int, int>();
        foreach (int num in arr)
        {
            if (freq.ContainsKey(num)) freq[num]++;
            else freq[num] = 1;
        }

        Array.Sort(arr, (a, b) =>
        {
            if (freq[b] != freq[a]) return freq[b] - freq[a];
            return a - b;
        });
        Console.WriteLine("[" + string.Join(", ", arr) + "]");
    }
}
```

10.

```
using System;
using System.Linq;

class Program
{
    static void Main()
    {
        var words = Console.ReadLine().Split();
        var grouped = words.GroupBy(w => new string(w.OrderBy(c => c).ToArray()))
                           .Select(g => g.ToList())
                           .ToList();

        Console.WriteLine("[" + string.Join(", ", grouped.Select(g => "[" +
string.Join(", ", g) + "]")) + "]");
    }
}
```

11.

```
using System;

class Program
{
    static void Main()
    {
        string s = Console.ReadLine();
        if (string.IsNullOrEmpty(s))
        {
            Console.WriteLine("");
            return;
        }

        int start = 0, maxLength = 1;

        for (int i = 0; i < s.Length; i++)
        {
            // Odd length palindrome
            int l = i, r = i;
            while (l >= 0 && r < s.Length && s[l] == s[r])
            {
                if (r - l + 1 > maxLength)
                {
                    start = l;
                    maxLength = r - l + 1;
                }
                l--;
                r++;
            }

            // Even length palindrome
            l = i; r = i + 1;
            while (l >= 0 && r < s.Length && s[l] == s[r])
            {
                if (r - l + 1 > maxLength)
                {
                    start = l;
                    maxLength = r - l + 1;
                }
                l--;
                r++;
            }
        }

        Console.WriteLine(s.Substring(start, maxLength));
    }
}
```

12.

```
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Read array input
        string input = Console.ReadLine();
        int[] nums = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

        // Read lower and upper bounds
        int lower = int.Parse(Console.ReadLine());
        int upper = int.Parse(Console.ReadLine());

        List<string> result = new List<string>();
        int prev = lower - 1;
        for (int i = 0; i <= nums.Length; i++)
        {
            int curr = (i < nums.Length) ? nums[i] : upper + 1;

            if (curr - prev >= 2)
            {
                int start = prev + 1;
                int end = curr - 1;

                if (start == end)
                    result.Add(start.ToString());
                else
                    result.Add(start + "->" + end);
            }

            prev = curr;
        }

        // Print result in the requested format
        Console.Write("[");
        for (int i = 0; i < result.Count; i++)
        {
            Console.Write("\"" + result[i] + "\"");
            if (i < result.Count - 1)
                Console.Write(", ");
        }
        Console.WriteLine("]");
    }
}
```

13.

```
using System;

class Program
{
    static void Main()
    {
        // Read input array
        string input = Console.ReadLine();
        int[] nums = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

        if (nums.Length == 0)
        {
            Console.WriteLine("-1");
            return;
        }

        int n = nums.Length;

        // If only one element or first element is peak
        if (n == 1 || nums[0] > nums[1])
        {
            Console.WriteLine(0);
            return;
        }

        // If last element is peak
        if (nums[n - 1] > nums[n - 2])
        {
            Console.WriteLine(n - 1);
            return;
        }

        // Check middle elements
        for (int i = 1; i < n - 1; i++)
        {
            if (nums[i] > nums[i - 1] && nums[i] > nums[i + 1])
            {
                Console.WriteLine(i);
                return;
            }
        }

        Console.WriteLine("-1"); // Should never happen if array has at least one
peak
    }
}
```

14.

```
using System;

class Program
{
    static void Main()
    {
        // Read input array
        string input = Console.ReadLine();
        int[] arr = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

        // Read k value
        int k = int.Parse(Console.ReadLine());

        // Sort array in descending order
        Array.Sort(arr);
        Array.Reverse(arr);

        // Print kth largest element (1-based index)
        if (k > 0 && k <= arr.Length)
            Console.WriteLine(arr[k - 1]);
        else
            Console.WriteLine("Invalid");
    }
}
```

15.

```
using System;

class Program
{
    static void Main()
    {
        // Read rows and columns
        string[] parts = Console.ReadLine().Split();
        int rows = int.Parse(parts[0]);
        int cols = int.Parse(parts[1]);

        // Read matrix
        int[,] a = new int[rows, cols];
        for (int i = 0; i < rows; i++)
        {
            int[] row = Array.ConvertAll(Console.ReadLine().Split(), int.Parse);
            for (int j = 0; j < cols; j++)
                a[i, j] = row[j];
        }

        int top = 0, bottom = rows - 1, left = 0, right = cols - 1;
        Console.Write("[");

        bool first = true;
        while (top <= bottom && left <= right)
        {
            for (int i = left; i <= right; i++) { if (!first) Console.Write(",");
Console.Write(a[top, i]); first = false; }
            top++;

            for (int i = top; i <= bottom; i++) { if (!first) Console.Write(",");
Console.Write(a[i, right]); first = false; }
            right--;

            if (top <= bottom)
            {
                for (int i = right; i >= left; i--) { if (!first)
Console.Write(","); Console.Write(a[bottom, i]); first = false; }
                bottom--;
            }

            if (left <= right)
            {
                for (int i = bottom; i >= top; i--) { if (!first)
Console.Write(","); Console.Write(a[i, left]); first = false; }
                left++;
            }
        }

        Console.WriteLine("]");
    }
}
```

16.

```
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        string input = Console.ReadLine();
        int[] arr = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

        List<int> result = new List<int>();
        HashSet<int> seen = new HashSet<int>();
        HashSet<int> added = new HashSet<int>();
        foreach (int num in arr)
        {
            if (seen.Contains(num) && !added.Contains(num))
            {
                result.Add(num);
                added.Add(num);
            }
            else
            {
                seen.Add(num);
            }
        }

        Console.Write("[");
        for (int i = 0; i < result.Count; i++)
        {
            Console.Write(result[i]);
            if (i < result.Count - 1)
                Console.Write(",");
        }
        Console.WriteLine("]");
    }
}
```

17.

```
using System;

class Program
{
    static void Main()
    {

        string[] words = Console.ReadLine().Split();

        if (words.Length == 0)
        {
            Console.WriteLine("");
            return;
        }

        string prefix = words[0];

        for (int i = 1; i < words.Length; i++)
        {
            while (!words[i].StartsWith(prefix))
            {
                prefix = prefix.Substring(0, prefix.Length - 1);
                if (prefix == "")
                {
                    Console.WriteLine("");
                    return;
                }
            }
        }

        Console.WriteLine(prefix);
    }
}
```

18.

```
using System;

class Program
{
    static void Main()
    {
        string s = Console.ReadLine();
        int n = s.Length;
        int count = 0;

        for (int center = 0; center < 2 * n - 1; center++)
        {
            int left = center / 2;
            int right = left + center % 2;

            while (left >= 0 && right < n && s[left] == s[right])
            {
                count++;
                left--;
                right++;
            }
        }

        Console.WriteLine(count);
    }
}
```

19.

```
using System;
using System.Collections.Generic;
class Program
{
static void Main()
{
string input = Console.ReadLine();
        int[] a = string.IsNullOrWhiteSpace(input) ? new int[0] :
Array.ConvertAll(input.Split(), int.Parse);

var list = new List<string>();
for (int i = 0; i < a.Length - 2; i++)
for (int j = i + 1; j < a.Length - 1; j++)
for (int k = j + 1; k < a.Length; k++)
if (a[i] + a[j] + a[k] == 0)
{
int[] t = { a[i], a[j], a[k] };
Array.Sort(t);
string triplet = $"[{t[0]},{t[1]},{t[2]}]";
if (!list.Contains(triplet)) list.Add(triplet);
}
Console.WriteLine("[" + string.Join(", ", list) + "]");

}

}
```

20.

```
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        string s = Console.ReadLine();
        string t = Console.ReadLine();
        if (string.IsNullOrEmpty(s) || string.IsNullOrEmpty(t)) {
Console.WriteLine(""); return; }

        Dictionary<char, int> tCount = new Dictionary<char, int>();
        foreach (char c in t) tCount[c] = tCount.ContainsKey(c) ? tCount[c] + 1 :
1;
```

```
        int left = 0, minLen = int.MaxValue, start = 0, formed = 0;
        Dictionary<char, int> window = new Dictionary<char, int>();

        for (int right = 0; right < s.Length; right++)
        {
            char c = s[right];
            window[c] = window.ContainsKey(c) ? window[c] + 1 : 1;

            if (tCount.ContainsKey(c) && window[c] == tCount[c]) formed++;

            while (formed == tCount.Count)
            {
                if (right - left + 1 < minLen) { minLen = right - left + 1; start
= left; }
                char l = s[left];
                window[l]--;
                if (tCount.ContainsKey(l) && window[l] < tCount[l]) formed--;
                left++;
            }
        }

        Console.WriteLine(minLen == int.MaxValue ? "" : s.Substring(start,
minLen));
    }
}
```

21.

```
using System;

class Program
{
    static void Main()
    {
        string s = Console.ReadLine();
        Permute(s.ToCharArray(), 0, s.Length - 1);
    }

    static void Permute(char[] arr, int l, int r)
    {
        if (l == r)
        {
            Console.WriteLine(new string(arr));
            return;
        }

        for (int i = l; i <= r; i++)
        {
            Swap(arr, l, i);
            Permute(arr, l + 1, r);
            Swap(arr, l, i); // backtrack
        }
    }

    static void Swap(char[] arr, int i, int j)
    {
        char temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }
}
```

22.

```
using System;
using System.Linq;
class Program
{
   static void Main()
   {
      Console.Write("Enter numbers: ");
       int[] nums = Console.ReadLine().Split().Select(int.Parse).ToArray();

       int n = nums.Length;
       int[] dp = new int[n];
       Array.Fill(dp, 1);

       for (int i = 0; i < n; i++)
           for (int j = 0; j < i; j++)
               if (nums[i] > nums[j])
                  dp[i] = Math.Max(dp[i], dp[j] + 1);

      Console.WriteLine("Length of LIS: " + dp.Max());
}
}
```

23.

```
using System;
class Program
{
   static void Main()
   {
      Console.Write("Enter numbers separated by spaces: ");
       string[] parts = Console.ReadLine().Split(' ');
       int n = parts.Length;
       int[] arr = new int[n];
       int[] result = new int[n];
       for (int i = 0; i < n; i++)
           arr[i] = int.Parse(parts[i]);
       for (int i = 0; i < n; i++)
       {
           result[i] = -1;
           for (int j = i + 1; j < n; j++)
           {
               if (arr[j] > arr[i])
               {
                  result[i] = arr[j];
                  break;
               }
           }
       }
      Console.Write("Next Greater Elements: ");
       for (int i = 0; i < n; i++)
          Console.Write(result[i] + " ");
   }
}
```

24.

```
using System;
class Program
{
   static void Main()
   {
      Console.Write("Enter numbers: ");
       string[] parts = Console.ReadLine().Split(' ');
       int n = parts.Length, limit = n / 3;
       int[] arr = new int[n];
       for (int i = 0; i < n; i++) arr[i] = int.Parse(parts[i]);

      Console.Write("Elements appearing more than n/3 times: ");
       for (int i = 0; i < n; i++)
```

```
        {
            int count = 0;
            for (int j = 0; j < n; j++) if (arr[j] == arr[i]) count++;
            bool dup = false;
            for (int k = 0; k < i; k++) if (arr[k] == arr[i]) dup = true;
            if (count > limit && !dup) Console.Write(arr[i] + " ");
        }
    }
}
```

25.

```
using System;
using System.Collections.Generic;

class Program
{
    static List<List<int>> result = new List<List<int>>();
    static void Main()
    {
        // Read input
        int[] nums = Array.ConvertAll(Console.ReadLine().Split(), int.Parse);
        int target = int.Parse(Console.ReadLine());
        Array.Sort(nums);
        Backtrack(nums, target, new List<int>(), 0);

        // Print result
        Console.Write("[");
        for (int i = 0; i < result.Count; i++)
        {
            Console.Write("[" + string.Join(",", result[i]) + "]");
            if (i < result.Count - 1) Console.Write(",");
        }
        Console.WriteLine("]");
    }

    static void Backtrack(int[] nums, int remain, List<int> current, int start)
    {
        if (remain == 0)
        {
            result.Add(new List<int>(current));
            return;
        }
        if (remain < 0) return;
        for (int i = start; i < nums.Length; i++)
        {
            current.Add(nums[i]);
            Backtrack(nums, remain - nums[i], current, i); // allow reuse
            current.RemoveAt(current.Count - 1);
        }
    }
}
```