class Program

{

    static void Main()

    {

        // Mapeo de los números del teclado a las letras correspondientes

        Dictionary<char, string> keypad = new Dictionary<char, string>

        {

            {'1', ""},          // "1" no tiene letras

            {'2', "ABC"},       // "2" tiene "A", "B", "C"

            {'3', "DEF"},       // "3" tiene "D", "E", "F"

            {'4', "GHI"},       // "4" tiene "G", "H", "I"

            {'5', "JKL"},       // "5" tiene "J", "K", "L"

            {'6', "MNO"},       // "6" tiene "M", "N", "O"

            {'7', "PQRS"},      // "7" tiene "P", "Q", "R", "S"

            {'8', "TUV"},       // "8" tiene "T", "U", "V"

            {'9', "WXYZ"},      // "9" tiene "W", "X", "Y", "Z"

            {'0', " "}          // "0" tiene espacio

        };

        // Mostrar mensajes

        Console.WriteLine("Introduce la secuencia de números.");

        Console.WriteLine("Usa '\*' para borrar un número y '#' para enviar la entrada.");

        Console.WriteLine("Ejemplo: '222 2 22' se convierte en 'CAB'.");

        // Bucle para permitir múltiples entradas, hasta que el usuario decida terminar

        while (true)

        {

            // Leer la secuencia de entradas del usuario

            string input = Console.ReadLine();

            // Lista para almacenar la cadena de resultado

            StringBuilder result = new StringBuilder();

            // Variable para almacenar la secuencia de números actual

            StringBuilder currentSequence = new StringBuilder();

            foreach (char c in input)

            {

                if (c == '#')

                {

                    // Cuando encontramos '#', procesamos la secuencia actual

                    ProcessAndDisplayResult(currentSequence.ToString(), keypad, result);

                    break;  // Terminamos cuando encontramos '#'

                }

                else if (c == '\*')

                {

                    // Si encontramos '\*', eliminamos el último número de la secuencia

                    if (currentSequence.Length > 0)

                    {

                        currentSequence.Remove(currentSequence.Length - 1, 1); // Eliminar el último número ingresado

                    }

                }

                else

                {

                    // Agregar el carácter a la secuencia

                    currentSequence.Append(c);

                }

            }

            // Mostrar el resultado final

            Console.WriteLine("Resultado: " + result.ToString());

        }

    }

    // Función para procesar la secuencia de números y mostrar el resultado

    static void ProcessAndDisplayResult(string sequence, Dictionary<char, string> keypad, StringBuilder result)

    {

        char lastKey = '\0';  // Tecla anterior para manejar la pausa entre secuencias

        int count = 1;        // Contador de veces que se presiona una tecla consecutiva

        // Procesamos la secuencia filtrada de números

        for (int i = 0; i < sequence.Length; i++)

        {

            char currentKey = sequence[i];

            if (currentKey == lastKey)

            {

                // Si es la misma tecla que la anterior, incrementamos el contador

                count++;

            }

            else

            {

                // Si es una tecla diferente, procesamos la tecla anterior

                if (lastKey != '\0')

                {

                    // Añadir el caracter correspondiente del mapeo

                    result.Append(GetCharacterFromKey(lastKey, count, keypad));

                }

                // Resetear el contador para la nueva tecla

                count = 1;  // Reiniciamos el contador al comenzar una nueva secuencia de tecla

            }

            lastKey = currentKey;

        }

        // Procesar la última tecla después de salir del bucle

        if (lastKey != '\0')

        {

            result.Append(GetCharacterFromKey(lastKey, count, keypad));

        }

    }

    // Función para obtener el caracter correspondiente a una tecla y cuántas veces se presiona

    static char GetCharacterFromKey(char key, int count, Dictionary<char, string> keypad)

    {

        // Si el key no existe en el diccionario, no procesar

        if (!keypad.ContainsKey(key))

        {

            return '\0';

        }

        string characters = keypad[key];

        if (characters.Length == 0)

        {

            return '\0';  // Retorna un valor nulo si no hay letras asignadas a esa tecla

        }

        // Ajustar el índice usando el módulo para que el contador "recicle" si es mayor que el número de letras disponibles

        int index = (count - 1) % characters.Length;

        return characters[index];

    }

}