EJERCICIO 8

===PalabraReversa.h===

```
#pragma once
#include <iostream>
#include <string>
using namespace std;

class CadenaOps
{
private:
  void impresionRecursiva(const string &texto, int index);

public:
  void imprimirInverso(const string &texto);
};

void CadenaOps::impresionRecursiva(const string &texto, int index)
{
  if (index >= texto.size()) //CASO BASE: Cuando el índice supera el tamaño de la
cadena, se detiene la recursión.
    return;
  impresionRecursiva(texto, index + 1); //PASO RECURSIVO:Primero se llama
recursivamente con el siguiente índice.
  cout << texto[index];
}

void CadenaOps::imprimirInverso(const string &texto)
{
  impresionRecursiva(texto, 0); // Llamamos a la función recursiva comenzando desde
el primer índice
}

/* ¿CUÁNDO ESTO SERÍA INFINITO?
    La recursión se volvería infinita si el índice 'index' no se incrementa
correctamente,
    o si se omite el caso base (index >= texto.size()), lo que provocaría un ciclo
indefinido
    y un desbordamiento de pila.

    También podría fallar si se manipula el índice de forma incorrecta, generando
accesos fuera de rango.
  */

  /* ¿POR QUÉ ES UNA SOLUCIÓN NATURAL?
    La recursión permite recorrer la cadena hasta el final y luego imprimir al
regresar,
    lo que invierte el orden de forma implícita sin usar estructuras auxiliares. */
```

===main.cpp===

```
#include "PalabraReversa.h"
#include <iostream>
#include <string>
#include <locale.h>

using namespace std;

int main(){
  string entrada;
  cout<<"Ingrese una palabra o frase corta: ";
  getline(cin,entrada);

  CadenaOps palabra;

  cout<< "Texto invertido: ";

  palabra.imprimirInverso(entrada);

}
```