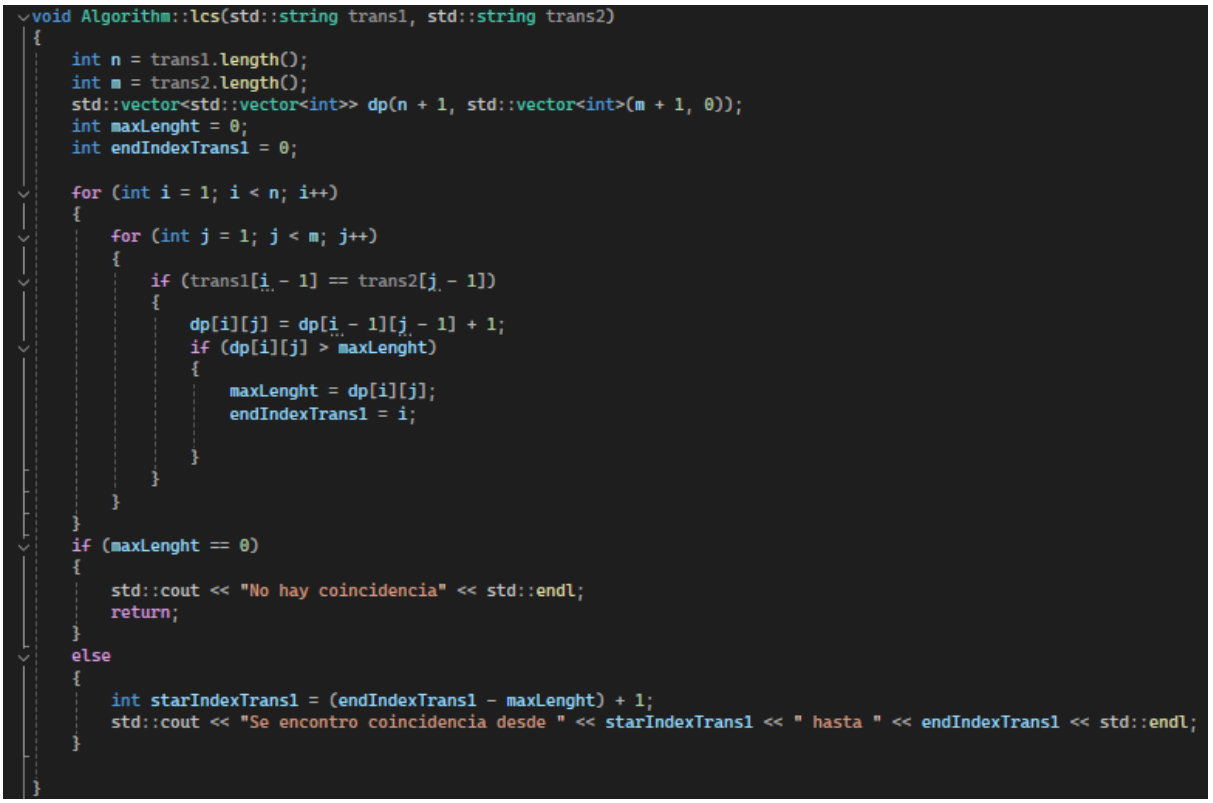
Rodrigo Rodriguez de Luna - A01384318

Reflexión

Durante esta actividad se utilizamos diferentes algoritmos de manejo de strings que hemos estudiado para poder resolver la actividad y donde identificamos la utilidad de cada uno de los para poder utilizarlos de la mejor manera al momento de resolver la actividad.Mi aportación en en esta actividad fue la parte 3 la cual consiste en comparar los dos archivos de transmisión, para poder encontrar y mostrar la posición inicial y la posición final del substring más largo que comparten.



```
void Algorithm::lcs(std::string trans1, std::string trans2)
{
    int n = trans1.length();
    int m = trans2.length();
    std::vector<std::vector<int>> dp(n + 1, std::vector<int>(m + 1, 0));
    int maxLenght = 0;
    int endIndexTrans1 = 0;

    for (int i = 1; i < n; i++)
    {
        for (int j = 1; j < m; j++)
        {
            if (trans1[i - 1] == trans2[j - 1])
            {
                dp[i][j] = dp[i - 1][j - 1] + 1;
                if (dp[i][j] > maxLenght)
                {
                    maxLenght = dp[i][j];
                    endIndexTrans1 = i;

                }
            }
        }
    }
    if (maxLenght == 0)
    {
        std::cout << "No hay coincidencia" << std::endl;
        return;
    }
    else
    {
        int starIndexTrans1 = (endIndexTrans1 - maxLenght) + 1;
        std::cout << "Se encontro coincidencia desde " << starIndexTrans1 << " hasta " << endIndexTrans1 << std::endl;
    }

}
```

Complejidad de la función lcs: O(n*m)

Esta función es el algoritmo de Longest Common Substring(LCS) y recibe de entrada los dos archivos de transmisión, para después regresar los valores de inicio y final del substring más largo.

Este empieza por definir sus valores iniciales necesarios para realizar su proceso, esto son definir la longitud de los archivos, tener variables para almacenar la longitud máxima de la subsecuencia común y el índice final. También se inicia un vector para crear una matriz que almacena los resultados intermedios del cálculo del LCS.

```cpp
void Algorithm::lcs(std::string trans1, std::string trans2)
{
    int n = trans1.length();
    int m = trans2.length();
    std::vector<std::vector<int>> dp(n + 1, std::vector<int>(m + 1, 0));
    int maxLenght = 0;
    int endIndexTrans1 = 0;
```

Después empieza el ciclo donde irá comparando cada una de las posibles combinaciones que puedan existir. cuando encuentra una coincidencia la anota en la matriz dp y en caso de que sea mayor que longitud máxima esta se vuelve la nueva longitud máxima y se actualiza el índice final.

```cpp
for (int i = 1; i < n; i++)
{
    for (int j = 1; j < m; j++)
    {
        if (trans1[i - 1] == trans2[j - 1])
        {
            dp[i][j] = dp[i - 1][j - 1] + 1;
            if (dp[i][j] > maxLenght)
            {
                maxLenght = dp[i][j];
                endIndexTrans1 = i;

            }
        }
    }
}
```

En caso de que la longitud máxima no se altere al final del ciclo, es resultado de que no hubo ninguna coincidencia y se imprime un mensaje acorde.

```cpp
if (maxLenght == 0)
{
    std::cout << "No hay coincidencia" << std::endl;
    return;
}
```

Cuando se encuentra una coincidencia al final del ciclo, se inicia una variable que es el índice final menos el tamaño de la coincidencia y da como resultado el inicio de coincidencia, que después se imprime junto al índice final, donde se encuentra esa coincidencia.

```cpp
else
{
    int starIndexTrans1 = (endIndexTrans1 - maxLenght) + 1;
    std::cout << "Se encontro coincidencia desde " << starIndexTrans1 << " hasta " << endIndexTrans1 << std::endl;
}
```