INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Teoría Computacional

Práctica 1: Operaciones con Lenguajes

Profesora: Luz María Sánchez García

Alumno: Mendoza Parra Sergio.

Grupo: 2CM1

Boleta: 2015630300

MEXICO, D.F. a 03 de Marzo del 2017

**Introducción:**

En este programa se verá el funcionamiento de operaciones con lenguajes. En este caso abran 8 operaciones con lenguajes los cuales primero se darán de forma teórica de como se llegó a la solución y posteriormente se visualizará en el programa el funcionamiento del mismo.

También se verán casos particulares como en las potencias negativas o inclusive con potencia 0, dependiendo del caso.

**Planteamiento del Problema:**

Haciendo uso de las expresiones regulares realizar un programa que calcule las siguientes operaciones con **n cadenas** de un determinado alfabeto ∑ (trabajar mínimo con 2 cadenas), en donde el usuario podrá seleccionar con cuales cadenas va a trabajar y qué tipo de operación va a seleccionar:

1) Determinar si es palíndromo

2) Calculo de Longitud

3) Concatenación de cadenas

4) Potencias (i>0, i<0, i=0)

5) Inverso o reflejo

6) Prefijos

7) Sufijos

8) Sub-cadenas

**Diseño y funcionamiento de la solución:**

**Determinación si es Palíndromo.**

Primero que nada tuve que verificar y corroborar que tipo de palíndromos hay, en este caso son palíndromo sin espacios, ya que si ponemos: “anita lava la tina” el programa llegaría a tener problemas con evaluar ese tipo de frase palíndroma.

Procedí a hacer una función la cual copiara en una cadena auxiliar, de manera inversa la cadena a verificar, en este caso lo hice para después compararla con la cadena original y si las cadenas son iguales, la función retornaría si, sí eran palíndromas o no. De este mismo modo si la cadena no es igual a la cadena original retornaría que no es una cadena palíndroma (ver el ejemplo).

Ejemplo:

Cadena Original

|  |  |  |
| --- | --- | --- |
| O | R | O |

Copia en una cadena “Auxiliar” de forma inversa

|  |  |  |
| --- | --- | --- |
| O | R | O |

En este ejemplo retorna, sí son palíndromos, ya que se leen igual se izquierda a derecha.

Ejemplo:

Cadena Original

|  |  |  |
| --- | --- | --- |
| C | A | T |

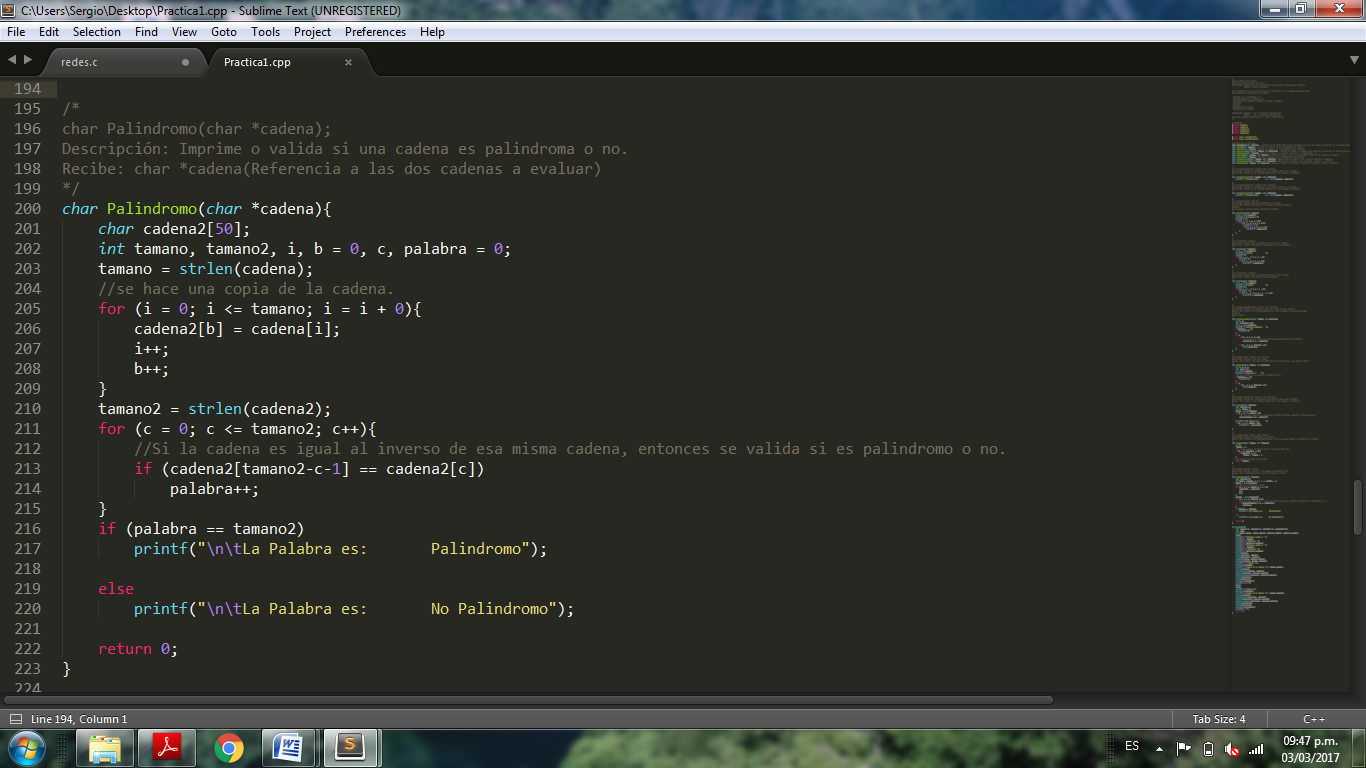
Copia en una cadena “Auxiliar” de forma inversa

|  |  |  |
| --- | --- | --- |
| T | A | C |

En este caso retorna que no es palíndromo ya que como se ve las palabras no son palíndromas.

**Implementación de la solución:**

**Determinación si es Palíndromo.**



**Cálculo de la longitud.**

En este caso es determinar la longitud o cuantos caracteres tiene la palabra. En este caso lo que yo hice fue ciclar o con un while ir contando los caracteres hasta que la cadena sea diferente de ‘\0’, una vez que se encontraba con el ‘\0’ se iba a retornar el número de caracteres los cuales tiene la palabra.

Ejemplo: Cadena Original

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| H | O | L | A | ‘\0’ |

Cont = 4

Cont = 3

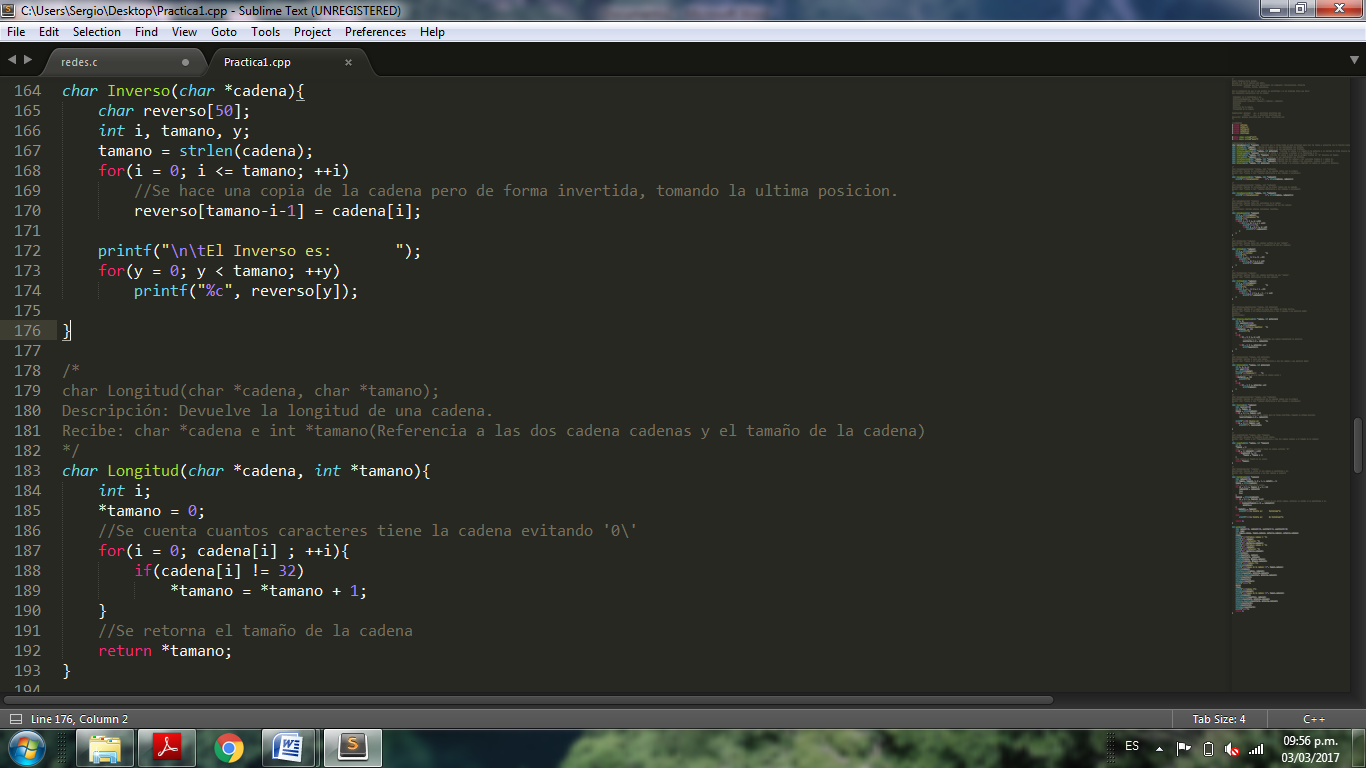
Cont = 2

Cont = 1

En este caso retorna 4 ya que después se tiene el carácter ‘\0’.

**Implementación de la solución:**

**Longitud de la cadena.**



**Concatenación de las Cadenas.**

En este caso lo que hice fue solo ingresar la segunda cadena a la cola de la primer cadena y de manera inversa hice lo mismo.

Ejemplo: Cadenas Originales

|  |  |  |  |
| --- | --- | --- | --- |
| H | O | L | A |
| L | O | L | A |

|  |
| --- |
| Cadena1 |

Cadena2

Concatenación: CADENA 1 + CADENA 2 || CADENA 2 + CADENA 1

Primer Caso:

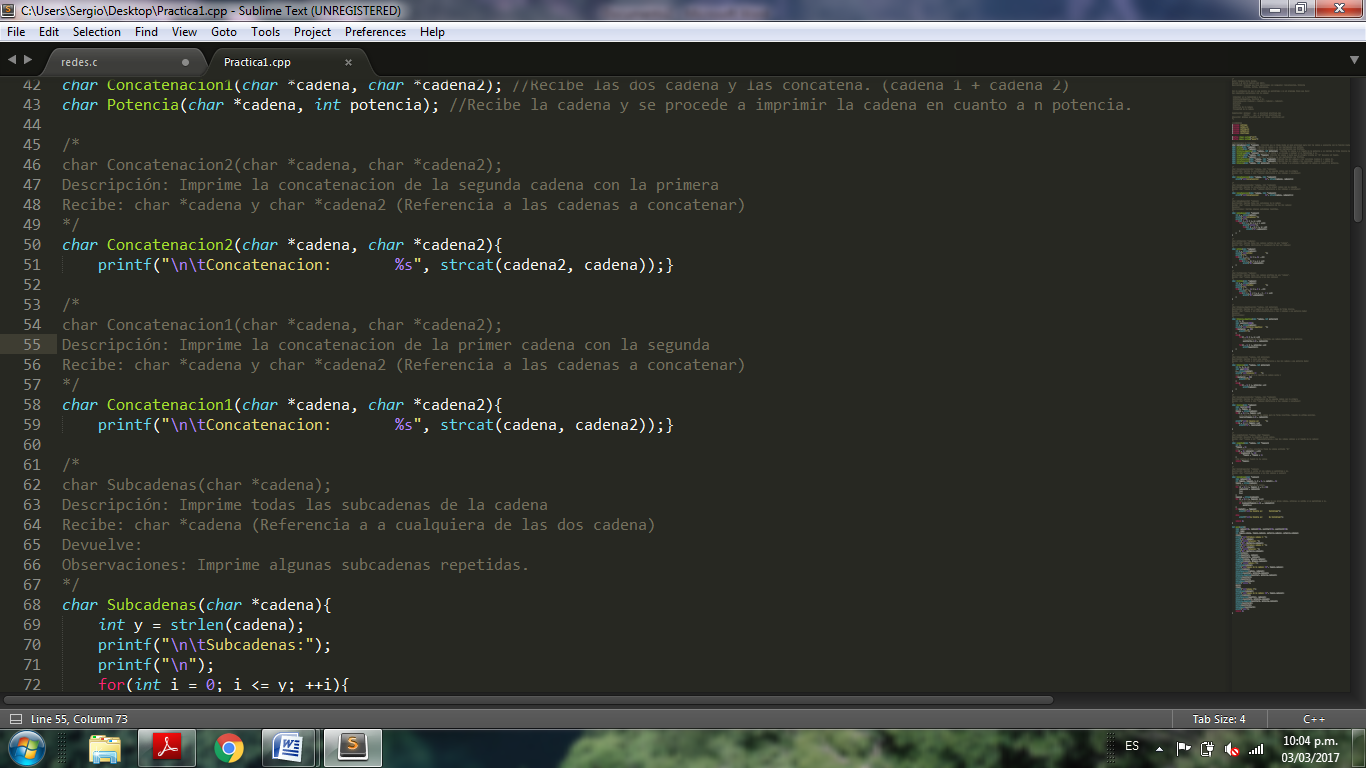
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| H | O | L | A | L | O | L | A |

Segundo Caso:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| L | O | L | A | H | O | L | A |

**Implementación de la solución:**

**Concatenación de cadenas.**

****

**Potencias (Positivas, Negativas o E)**

En este caso lo que hice fue hacer un ciclo en el cual dependiendo de la potencia se imprime varias veces una cadena de manera positiva, pero en el caso de la inversa lo copié en un arreglo auxiliar en el cual lo invertí y en la manera de potencia 0 solo se imprimió el carácter “E”.

Ejemplo: Cadena Positiva potencia = 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| H | O | L | A | H | O | L | A |

Cadena Negativa potencia = -2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | L | O | H | A | L | O | H |

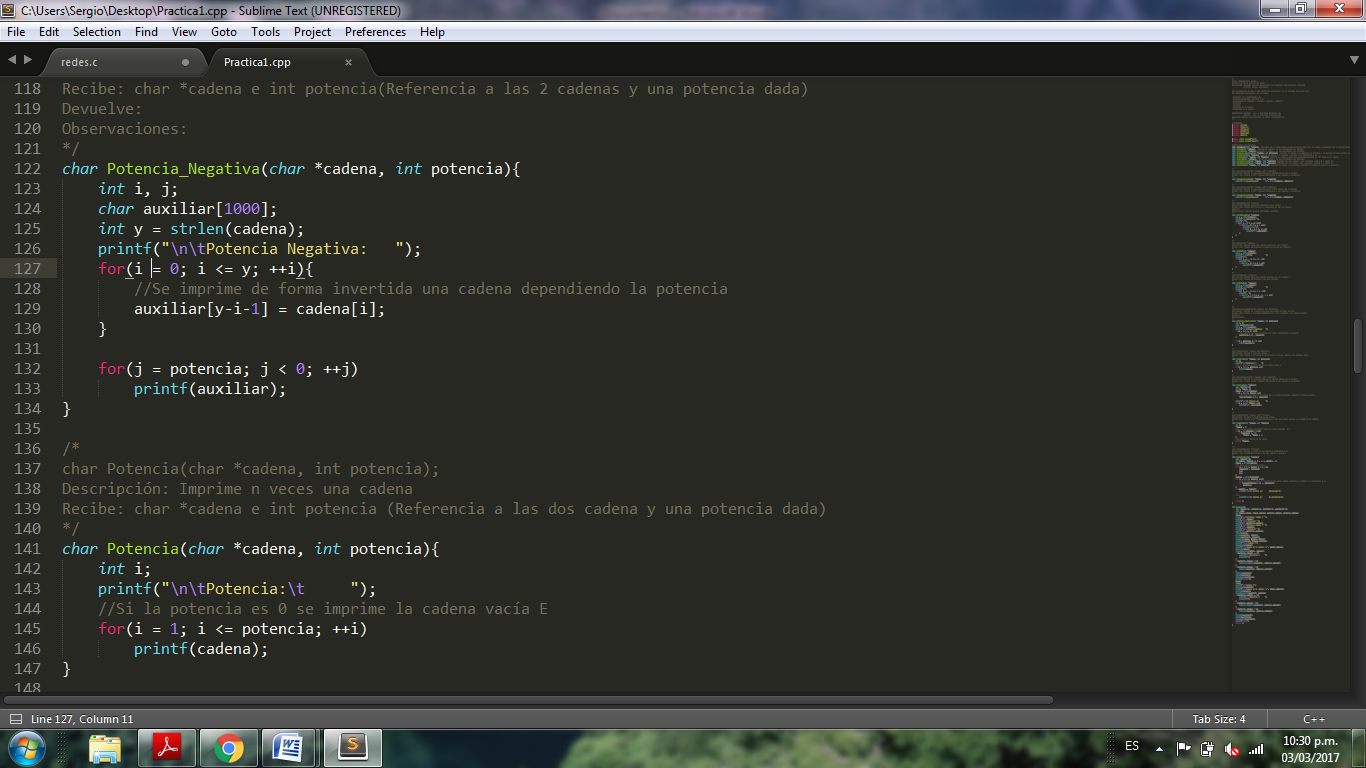
Cadena Vacía potencia = 0

|  |
| --- |
| E |

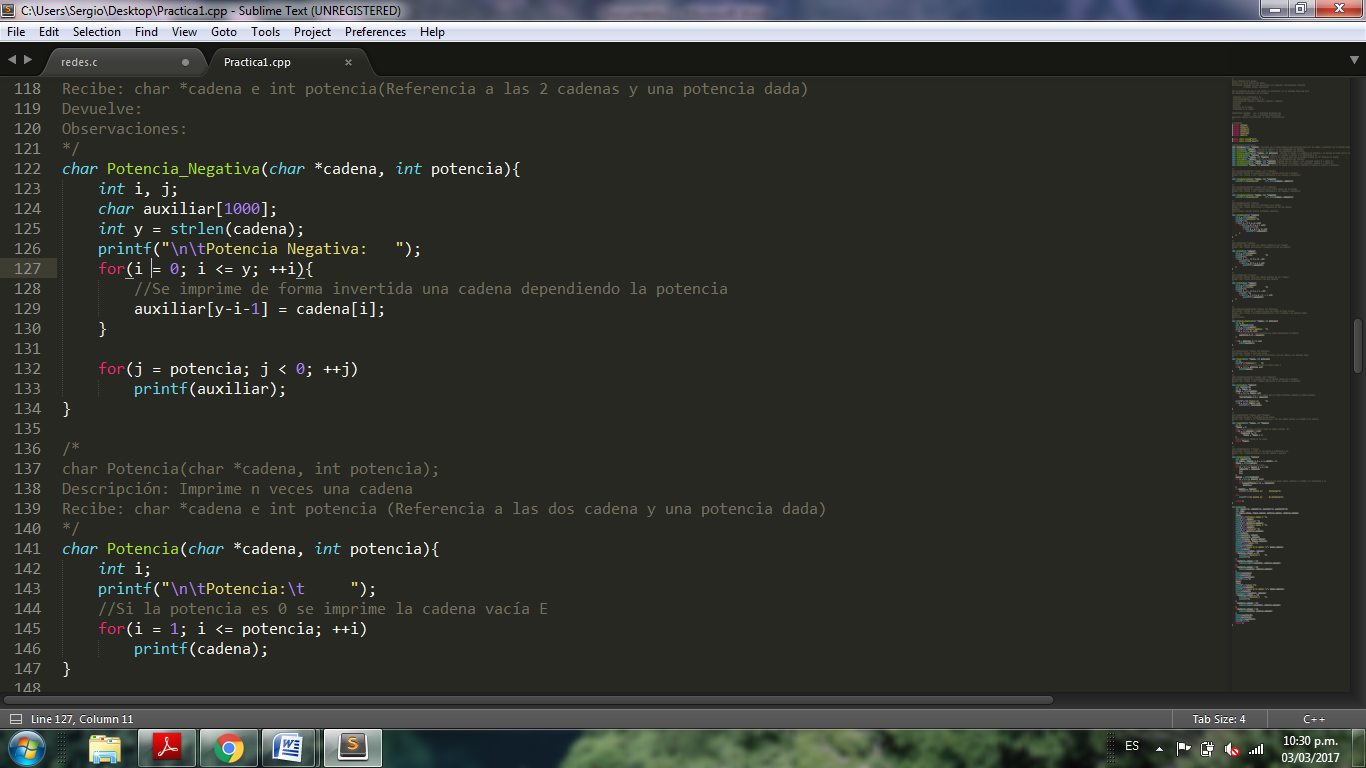
**Implementación de la solución:**

**Potencias (Negativa, Positiva, Vacía):**

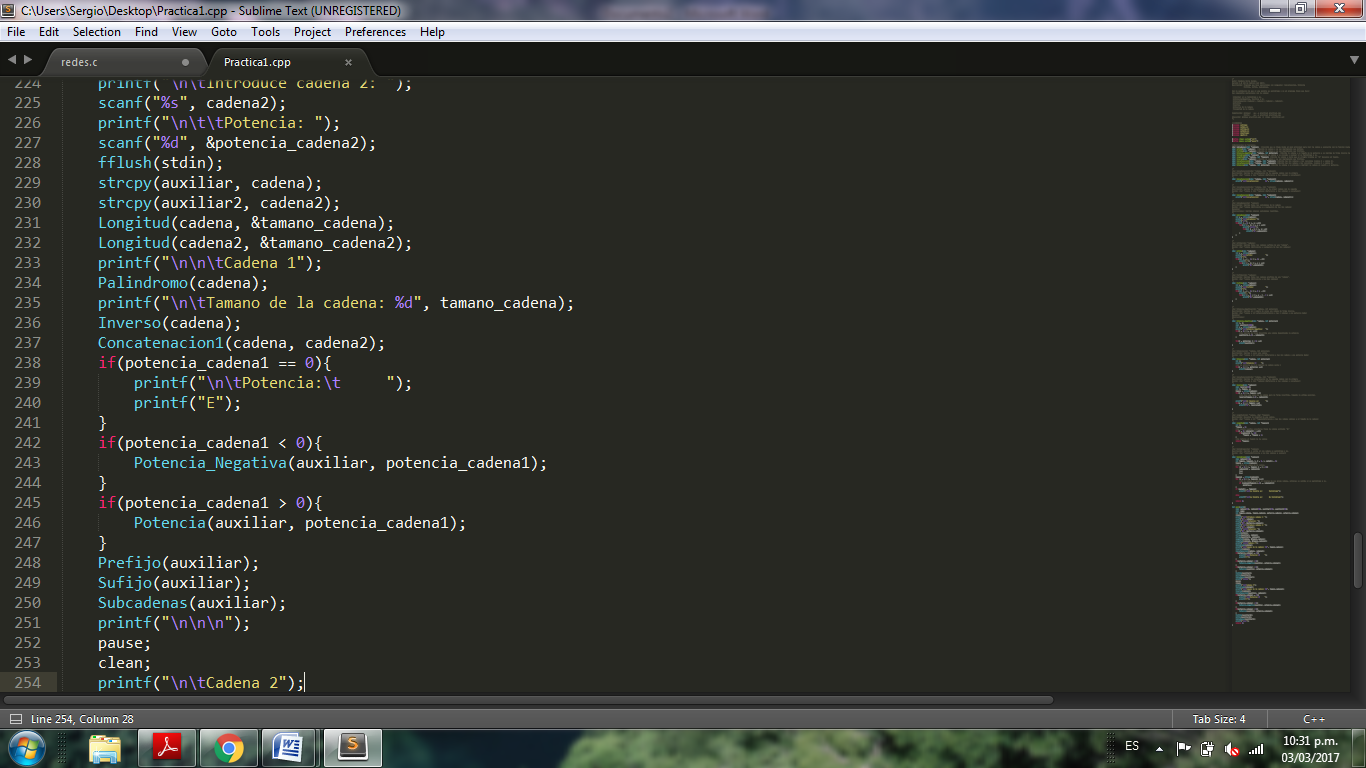
Función Potencia Negativa.

****

Función Potencia Positiva

****

Función Potencia Vacía

****

**Inverso o Reflejo**

En este caso lo que hice fue copiar la cadena en una cadena auxiliar de manera inversa y después lo imprimí de forma normal.

Ejemplo: Cadena Original

|  |  |  |  |
| --- | --- | --- | --- |
| H | O | L | A |

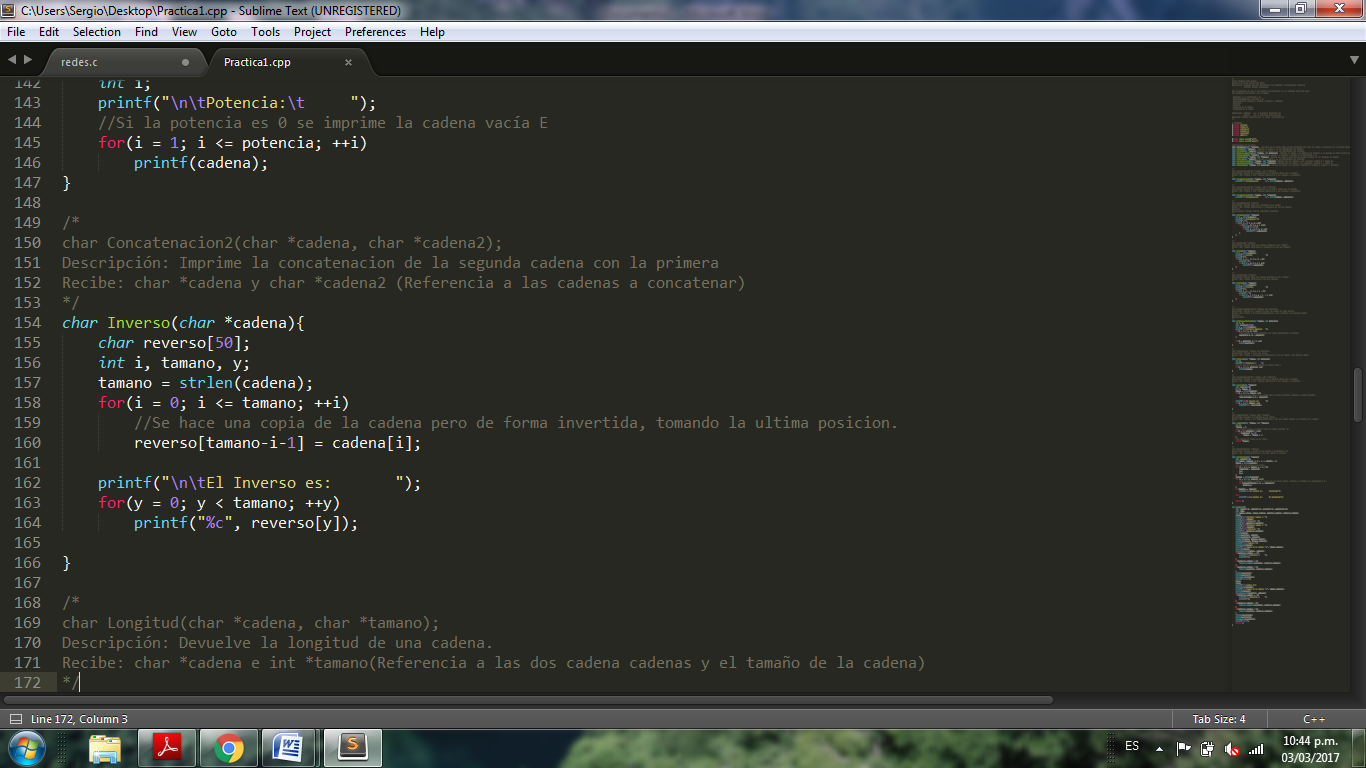
Inversión

|  |  |  |  |
| --- | --- | --- | --- |
| A | L | O | H |

De esta manera se mostrará el siguiente código que hace la función de invertir.

**Implementación de la solución:**

**Inversión de la cadena:**



**Prefijos**

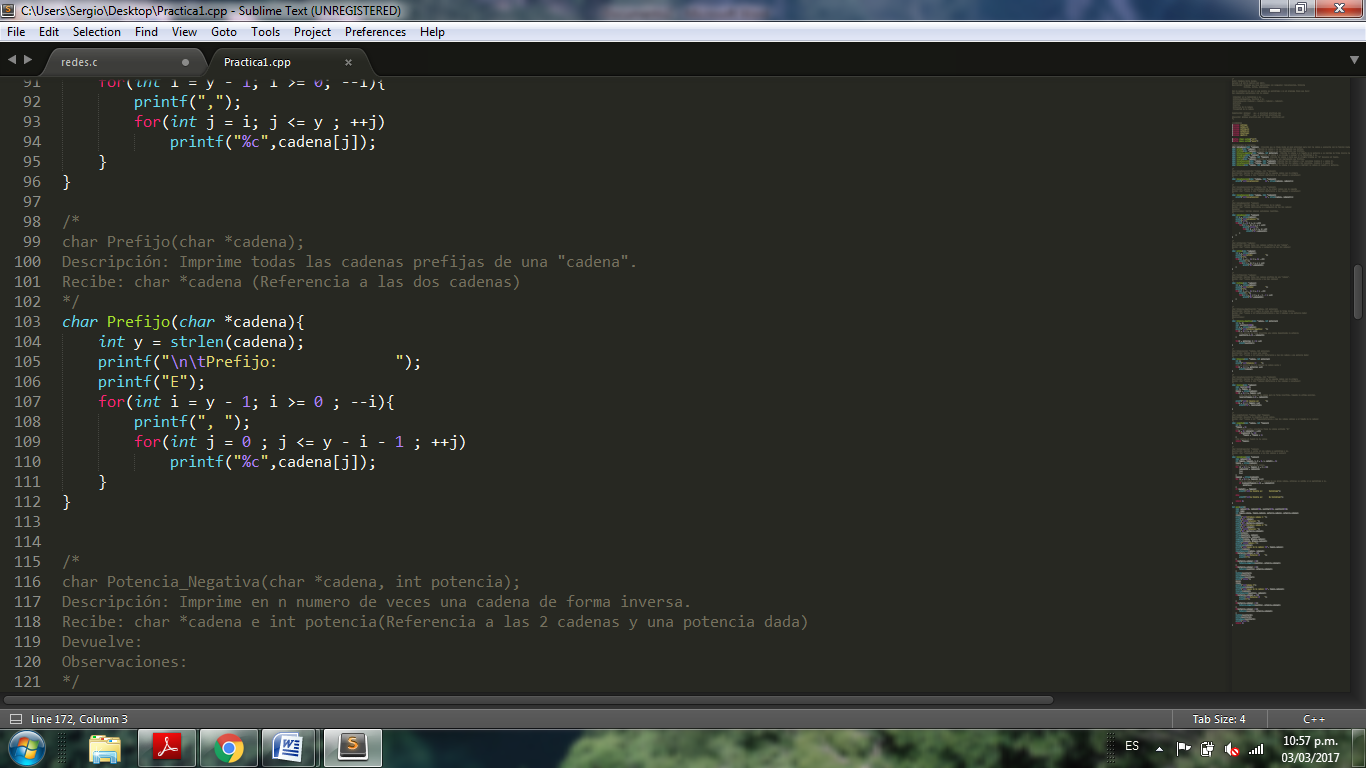
En este caso lo que hice fue hacer como una pirámide de las letras consecutivas de la cadena; por ejemplo:

Cadena = BBBBA Prefijo.

= {E, B, BB, BBB, BBBB, BBBBA}

**Implementación de la solución:**

**Prefijos:**



**Sufijos**

Este caso fue muy similar al de los prefijos solo que en este caso solo tenemos que imprimir el ultimo carácter de la cadena y posteriormente se tienen que imprimir las anteriores para que formen la cadena original.

Ejemplo:

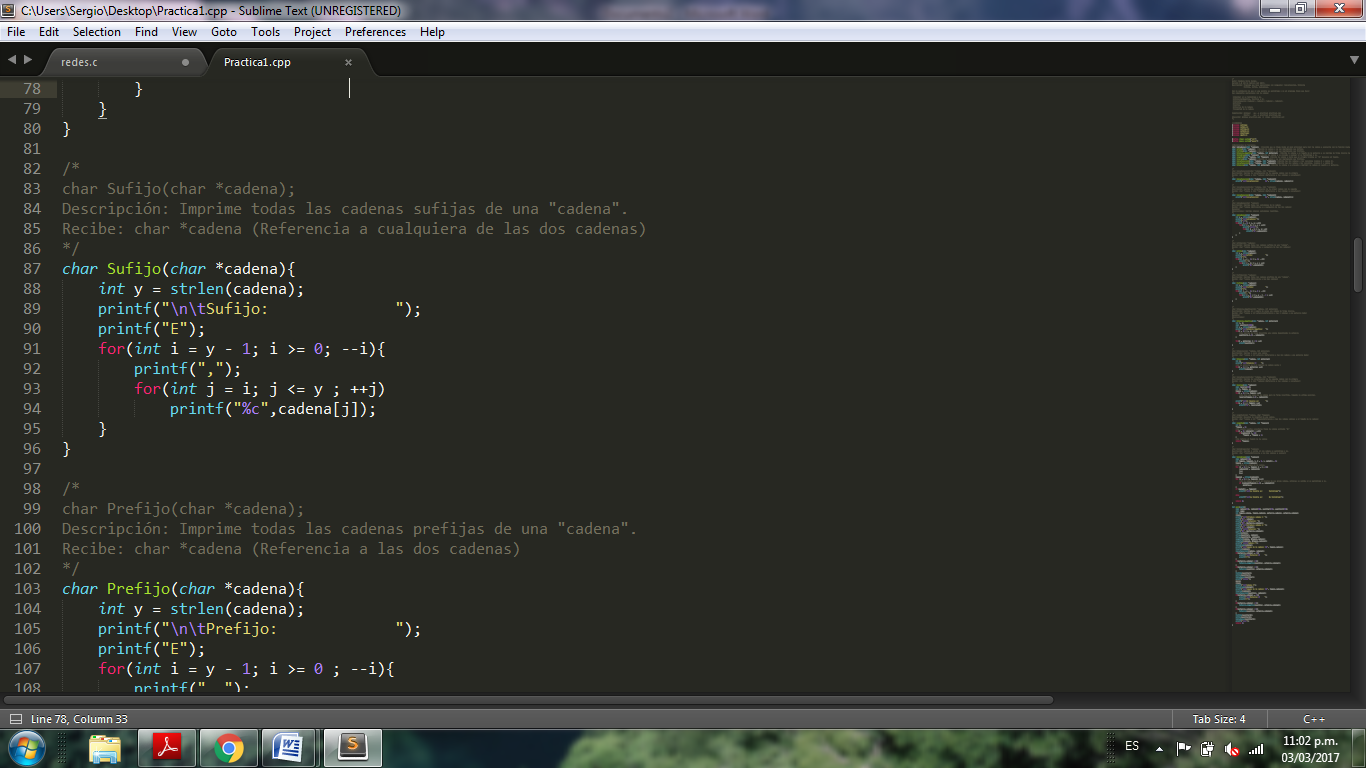
Cadena = HOLA  **SUFIJOS**

= {E, A, LA, OLA, HOLA}

Este caso se puede ver mejor con la siguiente función:

**Implementación de la solución:**

**Sufijos:**

****

**Sub-Cadenas**

Por último tenemos el caso de las sub-cadenas. Lo primero que hice fue analizar cómo se comportaban las sub-cadenas de este modo me di cuenta que al momento de imprimir las cadenas posteriormente se tenía que quitar el carácter de al principio y así sucesivamente de tal manera que solo quedara la letra final de la cadena.

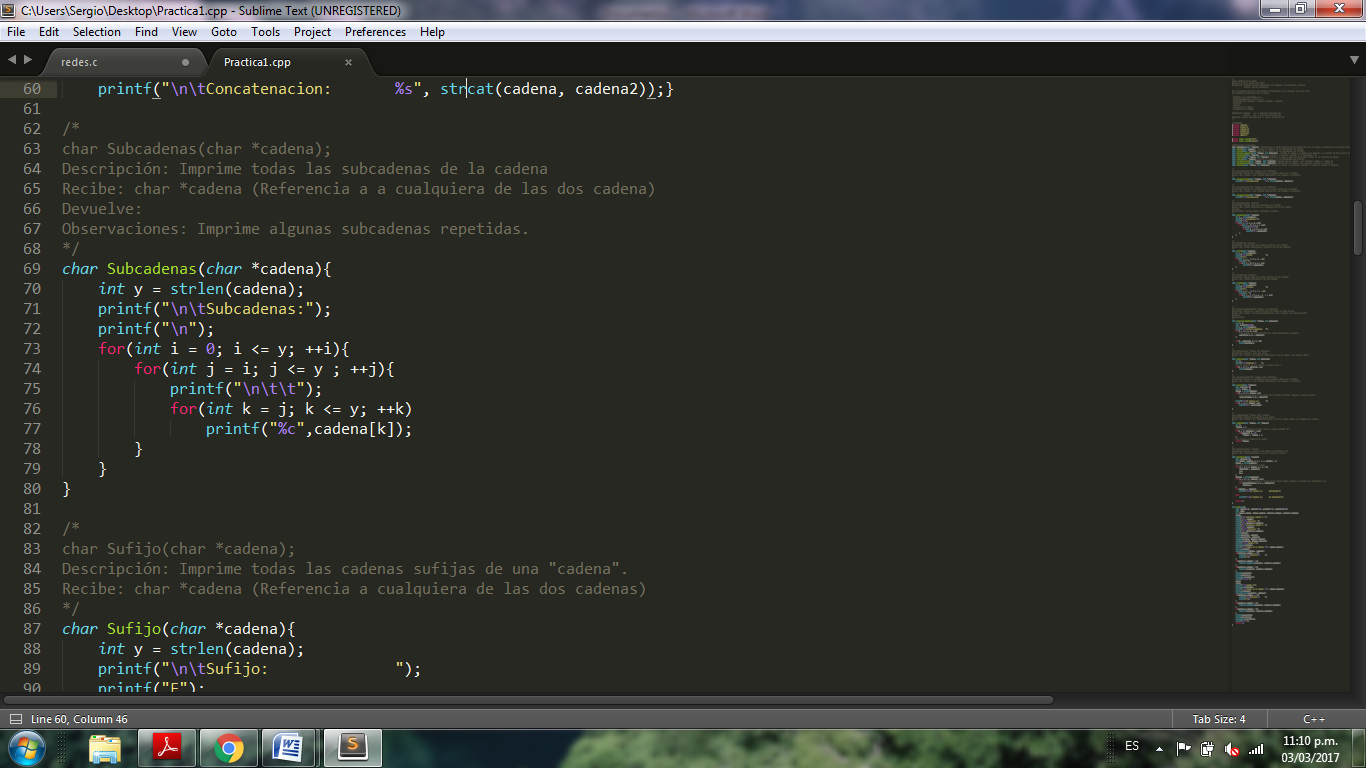
Ejemplo: Cadena = BBA

= {E, B, A, BA, AA, BB, AB, BBA}

De este modo se puede ver con la siguiente función la manera en cómo se imprimen las sub-cadenas.

**Implementación de la solución:**

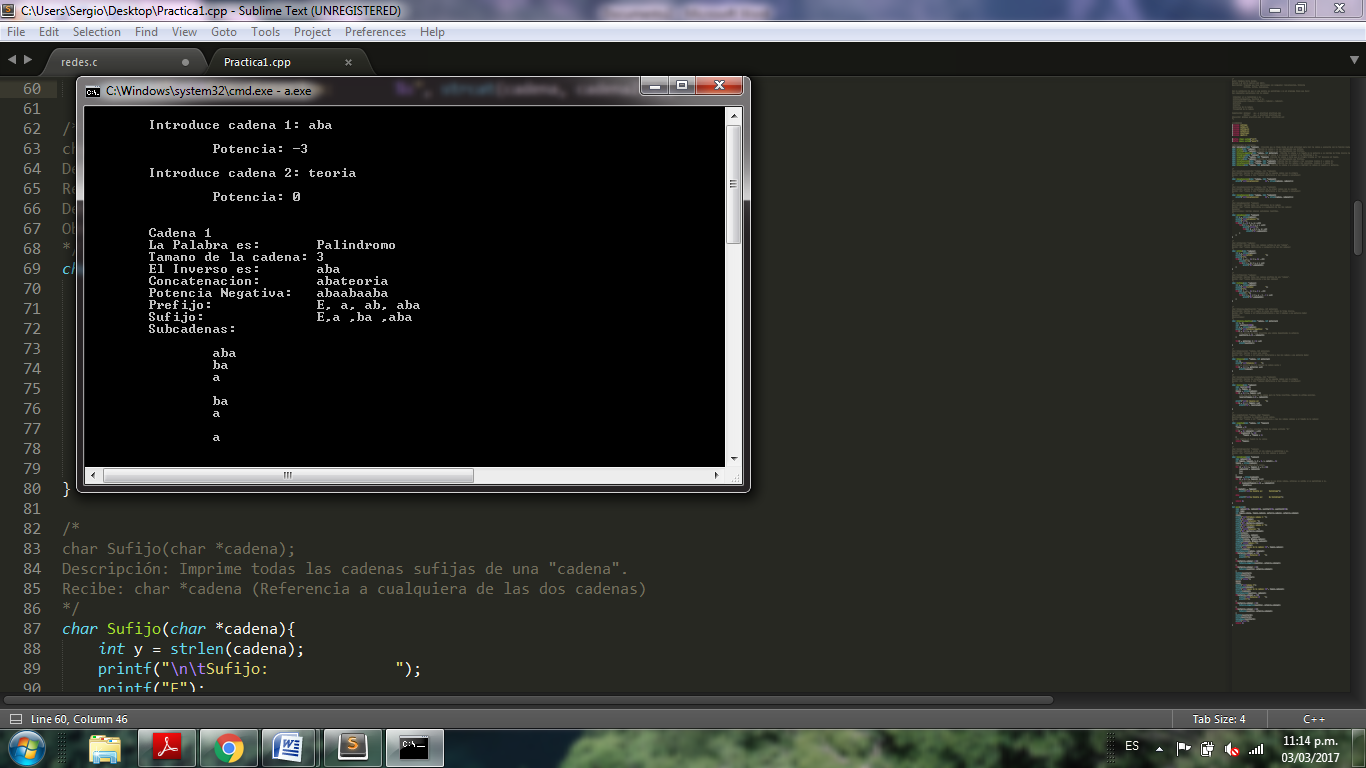
**Sub-cadenas:**



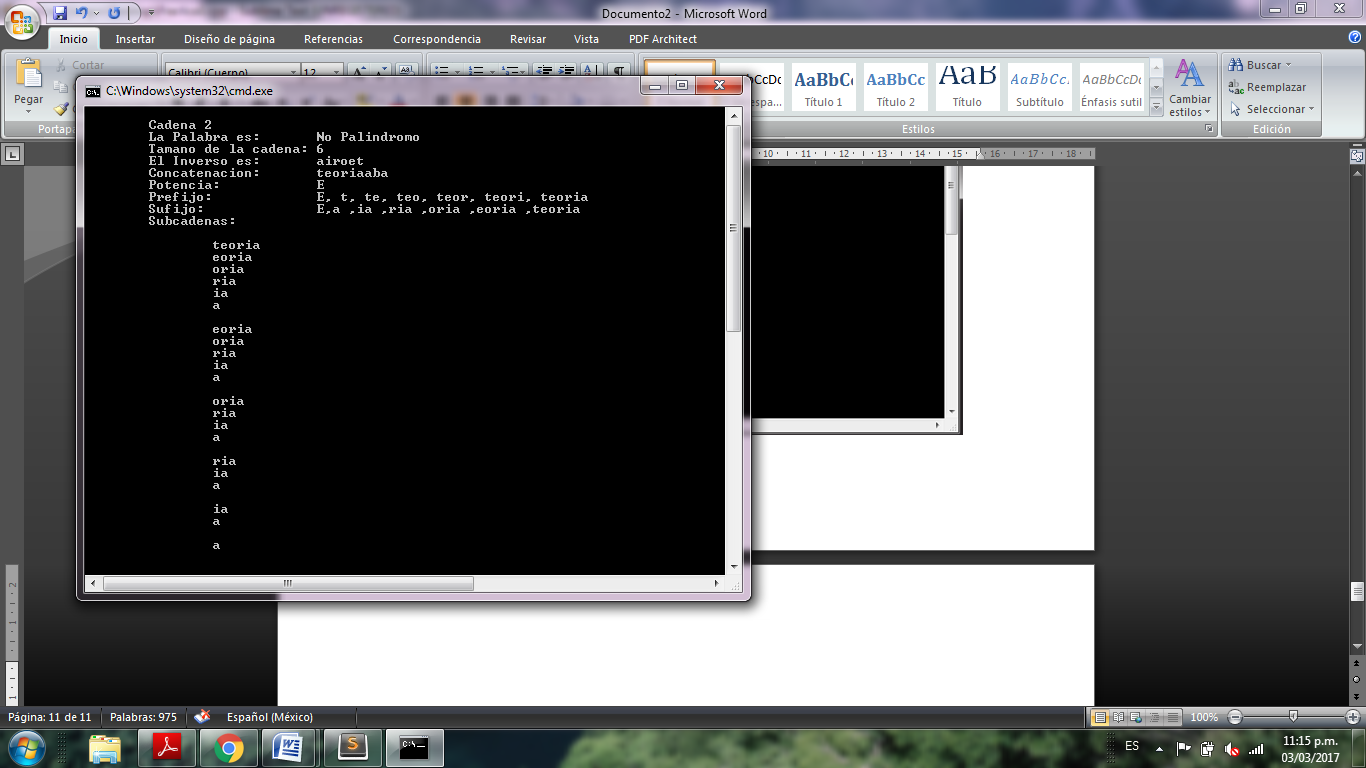
**Funcionamiento**

Una vez que programamos todo lo anterior procedemos a la compilación del programa en el cual vamos a tener dos cadenas de entradas a los cuales se les van a hacer las operaciones con cadenas o lenguajes. La siguiente imagen mostrara el funcionamiento del programa:

En la primera parte podemos observar que tenemos que ingresar 2 cadenas a las cuales les tenemos que dar la potencia y en la primera pantalla se verán las operaciones que se hacen con la cadena 1.



En la siguiente parte de la ventana, se puede observar que están las operaciones que se hacen a la cadena 2.



**Errores detectados**

1. Al momento de ingresar una frase palíndroma muere el programa: Esto es porque en el programa no verifique ese caso de frases palíndromas, entonces si se ingresa una frase cualquiera en el programa se visualizará un error.

**Posibles mejoras**

Como se mencionó anteriormente en la validación de las frases palíndromas podría declarar un “char” en el que se guarden los espacios en blanco y después de eso comparar la cadena sin espacios con la inversa de la misma y si son iguales, retornaría que son palíndromos y si no solo se retorna que no son palíndromos.

**Conclusión**

Esta práctica se me hizo interesante ya que pudimos corroborar muchos de los ejercicios que hicimos en clase con el programa y también una de las partes que me costó un poco de trabajo fue la primera parte de corroborar si eran palíndromos o no ya que el método que estaba usando no era el más factible para resolver el problema.