



INSTITUTO POLITÉCNICO NACIONAL

Escuela Superior de Cómputo

Ing. en sistemas computacionales



Nombre:

Josue Macias Castillo

Profesora:

Luz María Sánchez García

Grupo:

2CM1

Boleta:

2015301058

Materia:

Teoría Computacional

Fecha de entrega:

3 de febrero del 2017

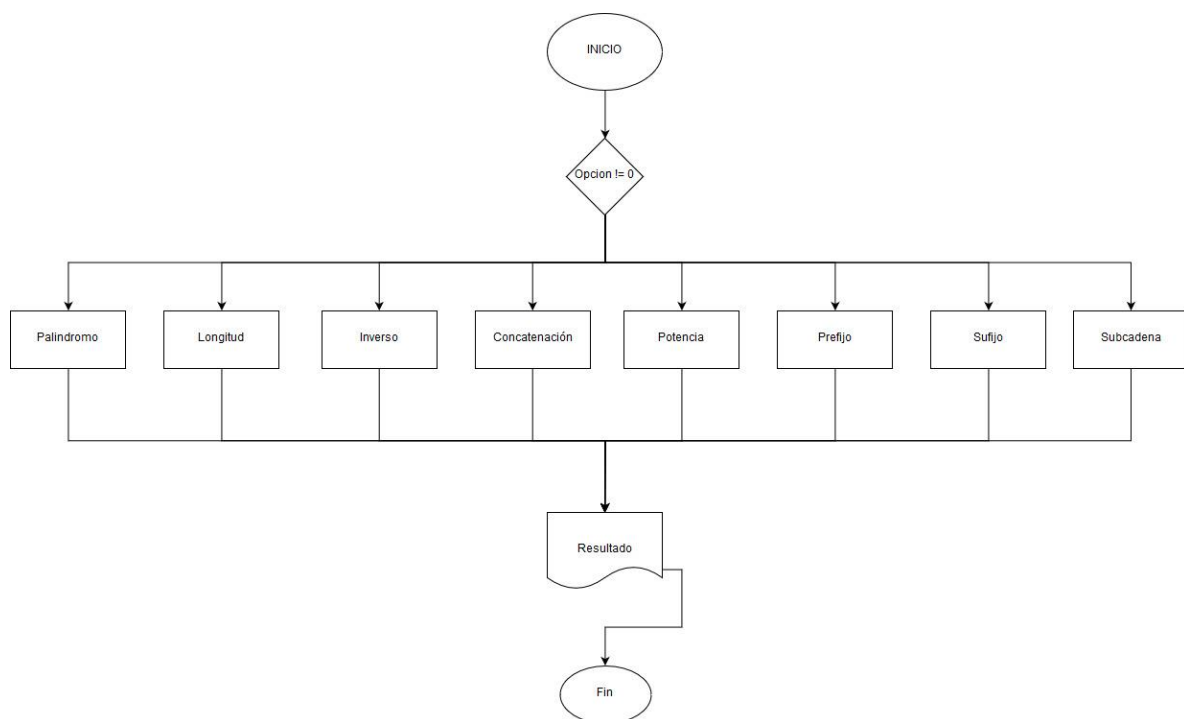
Introducción:

El propósito de este documento es reportar lo acontecido en la práctica número 1 de la unidad de aprendizaje teoría computacional, en la cual se nos pide que el usuario al menos ingrese 1 o 2 cadenas como mínimo para realizar las operaciones de cadenas como por ejemplo la concatenación, potencia, inverso, subcadenas entre otros, todo se muestra en un menú el cual da todas las opciones a elegir para cualquiera de las operaciones ya mencionadas.

Planteamiento del problema:

Se nos pide implementar una solución en C que nos muestre las operaciones de cadenas con ayuda de un menú para que sea más agradable al usuario, la solución se lleva a cabo con lo visto en la clase teórica.

Diseño de la solución:



Implementación de la solución:

```
/*
AUTOR: Josue Macias Castillo (C) Marzo 2017, 2CM1
VERSIÓN: 1.0

DESCRIPCIÓN: Programa que calcula las operaciones con cadenas

OBSERVACIONES:
El programa muestra un menu con todas las operaciones con cadenas
*/

//LIBRERIAS
#include "stdafx.h"
#include <iostream>
#include <string>
using namespace std;

//DECLARACIÓN DE FUNCIONES
/*Procedimiento para verificar si una palabra es palindroma
(Recibe un arreglo de cadena)*/
string Palindromo(string cadena);
/*Procedimiento para obtener longitud de una cadena
(Recibe un arreglo de cadena)*/
string Longitud(string cadena);
/*Procedimiento para invertir una cadena
(Recibe un arreglo de cadena y un arreglo auxiliar)*/
string Inverso(string cadena, string aux);
/*Procedimiento para concatenar 2 cadenas
(Recibe 2 arreglos de cadena)*/
string Concatenacion(string cadena, string cadena2);
/*Procedimiento para mostrar la potencia de una cadena
(Recibe un arreglo de cadena)*/
string Potencia(string cadena);
/*Procedimiento para mostrar el prefijo de una cadena
(Recibe un arreglo palabra)*/
string Prefijo(char palabra[50]);
/*Procedimiento para mostrar el sufijo de una cadena
(Recibe un arreglo palabra)*/
string Sufijo(char palabra[50]);
/*Procedimiento para mostrar las subcadenas de una cadena
(Recibe un arreglo palabra)*/
string Subcadena(char palabra[50]);

//PROGRAMA PRINCIPAL
int main()
{
    //Declaracion de variables
    int numero;
    char palabra[50];
    string cadena, cadena2, auxiliar;
    string subcadena;

    do
    {
        //Despliega las opciones para operar con el programa
```

```

cout << "1.- Palindromo" << endl;
cout << "2.- Longitud" << endl;
cout << "3.- Inverso" << endl;
cout << "4.- Concatenacion" << endl;
cout << "5.- Potencia" << endl;
cout << "6.- Prefijo" << endl;
cout << "7.- Sufijo" << endl;
cout << "8.- Subcadena" << endl;
cout << "0.- Salir" << endl;
cout << "Ingresa la opcion: ";
cin >> numero;

system("cls"); //Limpieza de pantalla
switch (numero)
{
case 1:

    Palindromo(cadena);

    break;
case 2:

    Longitud(cadena);

    break;
case 3:

    Inverso(cadena, auxiliar);

    break;
case 4:

    Concatenacion(cadena, cadena2);

    break;
case 5:

    Potencia(cadena);

    break;
case 6:

    Prefijo(palabra);

    break;
case 7:

    Sufijo(palabra);

    break;
case 8:

    Subcadena(palabra);

    break;
}

```

```

        system("pause");
        system("cls");
    } while (numero != 0);

    return 0;
}

/*
string Palindromo(string cadena);
Descripción: Procedimiento para verificar si una palabra es palindroma
Recibe: string cadena
Devuelve: Si en caso de ser palindroma, No en caso contrario
Observaciones: Al ingresar la cadena esta no debe contener espacios
*/
string Palindromo(string cadena)
{
    int op;
    int n;
    cout << "Escribe la cadena: ";
    cin >> cadena;

    n = cadena.length();

    for (int i = 0; i < n; i++)
    {
        if (cadena[i] == cadena[(n - i - 1)])
            op = 0;
        else
            op = 1;
    }
    if (op == 0)
        cout << "Es palindromo";
    if (op == 1)
        cout << "No es palindromo";
    cout << endl;

    return cadena;
}

/*
string Longitud(string cadena);
Descripción: Procedimiento para obtener longitud de una cadena
Recibe: string cadena
Devuelve: El tamaño de la cadena
Observaciones: Al ingresar la cadena esta no debe contener espacios
*/
string Longitud(string cadena)
{
    int n;

    cout << "Escribe la cadena: ";
    cin >> cadena;

    n = cadena.length();

    cout << "Su longitud es: " << n << endl;

```

```

        return string();
    }

    /*
    string Inverso(string cadena, string aux);
    Descripción: Procedimiento para obtener inverso de una cadena
    Recibe: string cadena, string aux
    Devuelve: El inverso de una cadena
    Observaciones: Al ingresar la cadena esta no debe contener espacios
    */
    string Inverso(string cadena, string aux)
    {
        int n;
        cout << "Escribe la cadena: ";
        cin >> cadena;

        n = cadena.length();
        for (int i = n; i >= 0; i--)
            aux += cadena[i];

        cout << "El inverso de la cadena es:" << aux << endl;

        return aux;
    }

    /*
    string Concatenacion(string cadena, string cadena2);
    Descripción: Procedimiento para concatenar 2 cadenas
    Recibe: string cadena, string cadena2
    Devuelve: La concatenacion de las cadenas
    Observaciones: Al ingresar cada cadena estas no deben contener espacios
    */
    string Concatenacion(string cadena, string cadena2)
    {
        string auxiliar;

        cout << "Escribe la primer cadena: ";
        cin >> cadena;
        cout << "Escribe la segunda cadena: ";
        cin >> cadena2;

        auxiliar = cadena + cadena2;

        cout << "La concatenacion es: " << auxiliar << endl;

        return auxiliar;
    }

    /*
    string Potencia(string cadena);
    Descripción: Procedimiento para obtener la potencia de una cadena
    Recibe: string cadena
    Devuelve: La potencia de una cadena
    Observaciones: Al ingresar la cadena esta no debe contener espacios
    */
    string Potencia(string cadena)
    {

```

```

    int x, n, m;
    char E = 156;
    string aux;

    cout << "Ingresa una cadena: ";
    cin >> cadena;

    cout << "Ingresa la potencia: ";
    cin >> x;

    n = cadena.length();

    if (x >= 1)
    {
        for (int i = 0; i < x; i++)
            cout << cadena;
    }
    else if (x < 0)
    {
        for (int i = n; i >= 0; i--)
            aux += cadena[i];

        m = x*(-1);
        for (int i = 0; i < m; i++)
            cout << aux;
    }
    else if (x == 0)
    {
        cout << E;
    }

    cout << endl;
    return string();
}

/*
string Prefijo(char palabra[50]);
Descripción: Procedimiento para obtener el prefijo de una cadena
Recibe: char palabra[50]
Devuelve: El prefijo de la cadena
Observaciones: Al ingresar la cadena esta no debe contener espacios
*/
string Prefijo(char palabra[50])
{
    int n;
    char E = 156;

    cout << "Ingresa una cadena: ";
    cin >> palabra;

    n = strlen(palabra);

    cout << E;
    for (int i = n; i >= 0; --i)
    {

```

```

        cout << endl;
        for (int j = 0; j <= n - i - 1; j++)
            cout << palabra[j];
    }
    cout << endl;
    return string();
}

/*
string Sufijo(char palabra[50]);
Descripción: Procedimiento para obtener el sufijo de una cadena
Recibe: char palabra[50]
Devuelve: El sufijo de la cadena
Observaciones: Al ingresar la cadena esta no debe contener espacios
*/
string Sufijo(char palabra[50])
{
    int n;
    char E = 156;

    cout << "Ingresa una cadena: ";
    cin >> palabra;

    n = strlen(palabra);

    cout << E;

    for (int i = n - 1; i >= 0; --i)
    {
        cout << endl;
        for (int j = i; j <= n; j++)
            cout << palabra[j];
    }
    cout << endl;
    return string();
}

/*
string Subcadena(char palabra[50]);
Descripción: Procedimiento para obtener las subcadenas de una cadena
Recibe: char palabra[50]
Devuelve: Las subcadenas de una cadena
Observaciones: Al ingresar la cadena esta no debe contener espacios
*/
string Subcadena(char palabra[50])
{
    int n;

    cout << "Ingresa una cadena: ";
    cin >> palabra;

    n = strlen(palabra);

    for (int i = 0; i <= n; i++)
    {
        for (int j = i; j <= n; j++)
        {

```



```

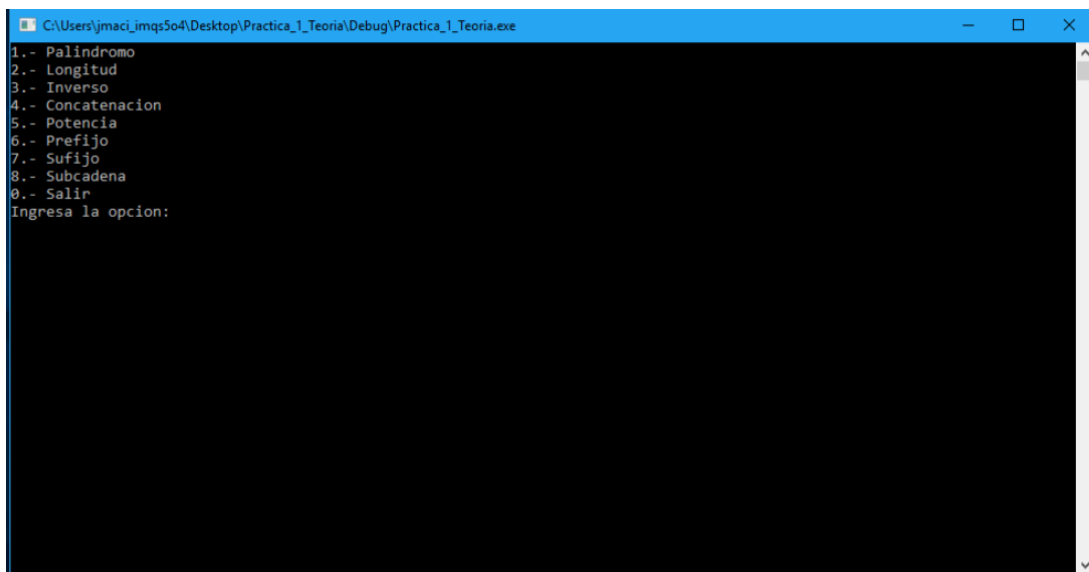
        cout << endl;
        for (int k = j; k <= n; k++)
            cout << palabra[k];
    }

    cout << endl;
    return string();
}

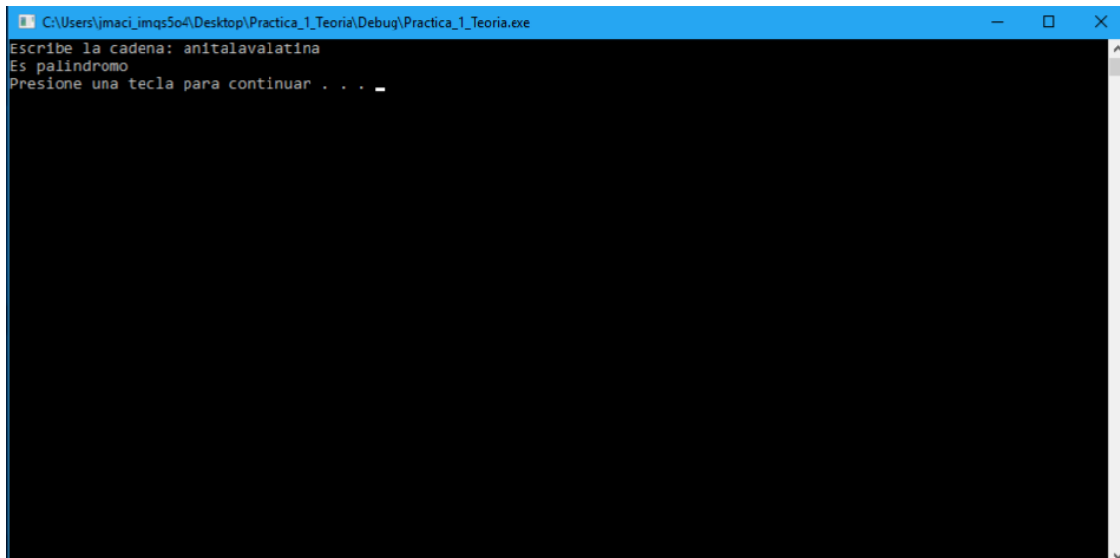
```

Funcionamiento:

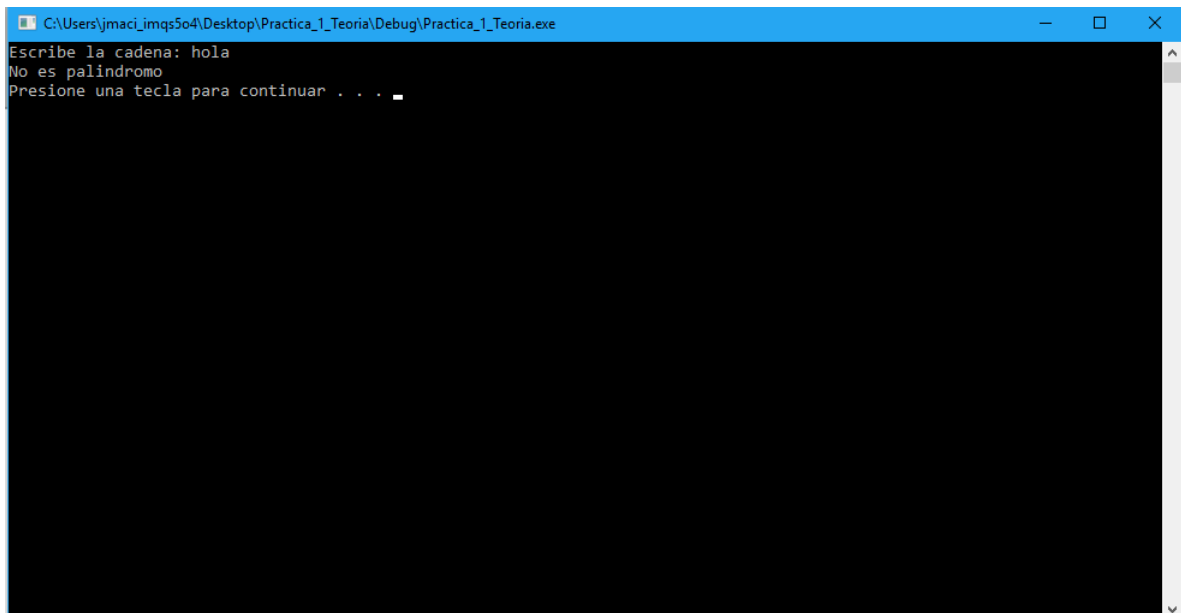
El programa se nos muestra de la siguiente forma



Primero ingresaremos a la opción 1, para determinar si una palabra es palíndroma

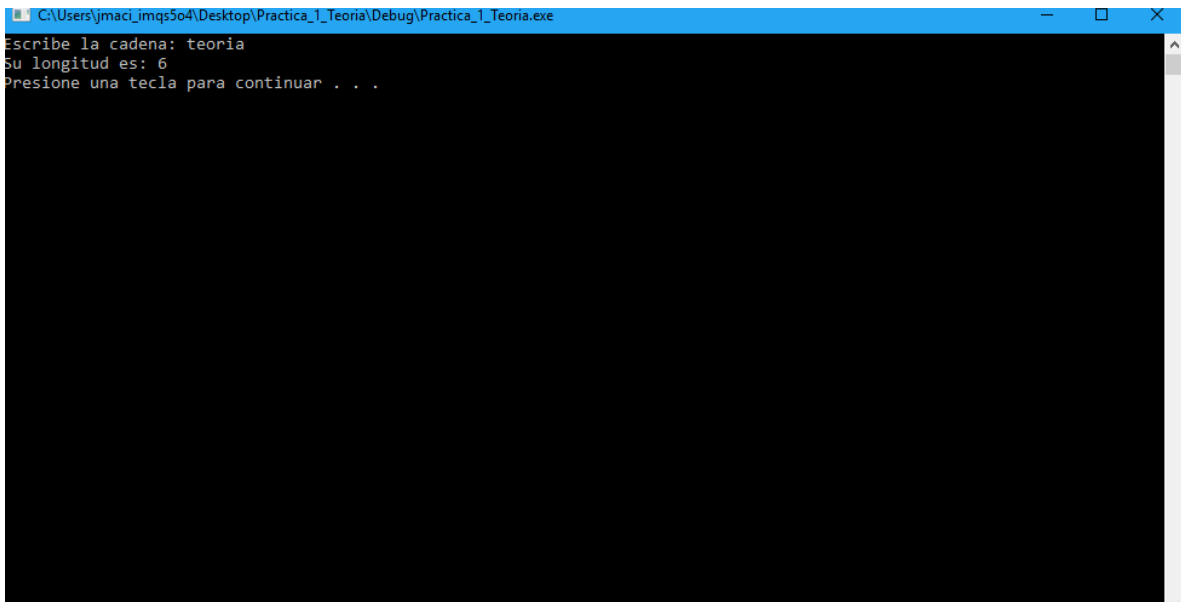


Ahora comprobemos con una no palíndroma



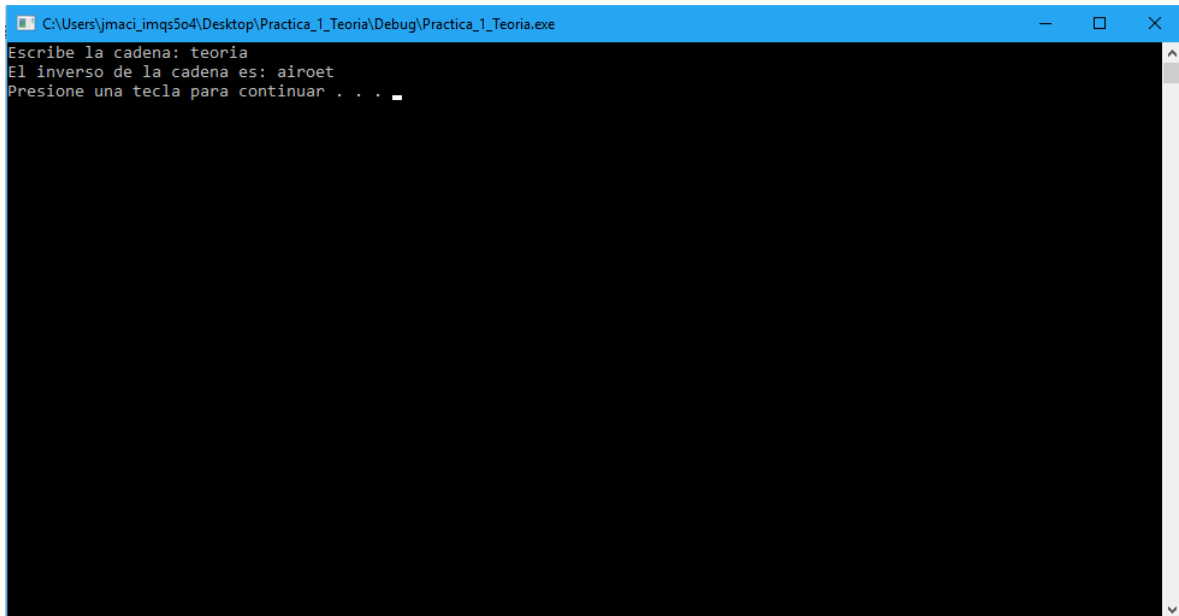
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Escribe la cadena: hola
No es palindromo
Presione una tecla para continuar . . . _
```

Regresando al menú la segunda opción es saber la longitud de una cadena, debemos tomar en cuenta que la cadena no debe tener espacios



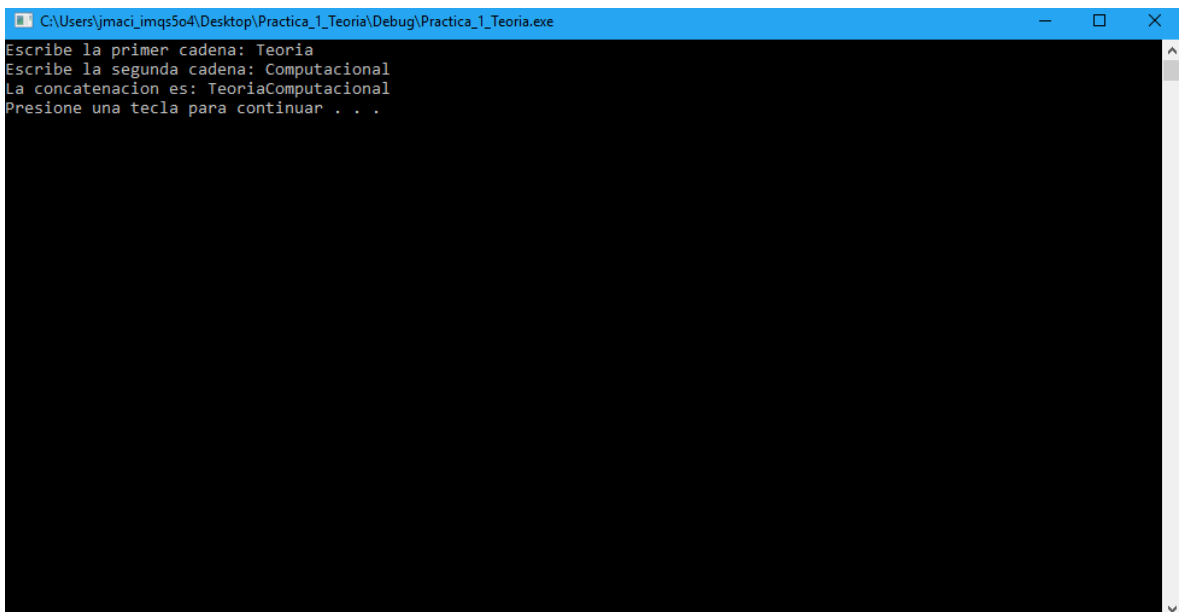
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Escribe la cadena: teoria
Su longitud es: 6
Presione una tecla para continuar . . .
```

La tercera opción muestra el inverso de una cadena, en este caso utilice la palabra teoría



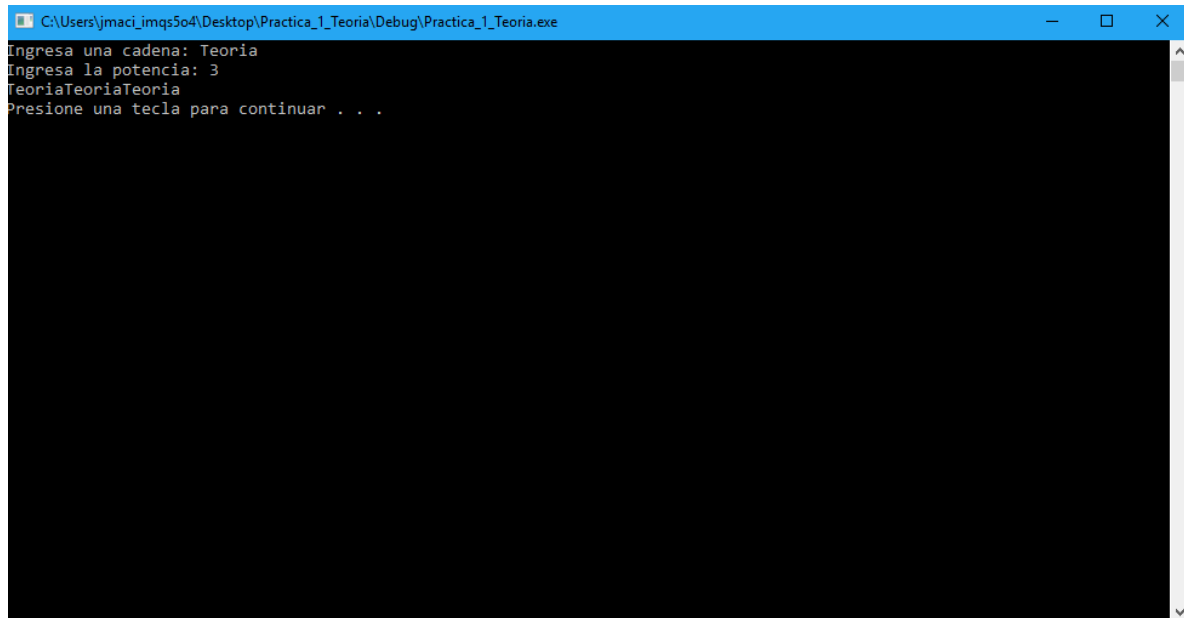
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Escribe la cadena: teoria
El inverso de la cadena es: airoet
Presione una tecla para continuar . . .
```

La cuarta opción muestra la concatenación de 2 cadenas



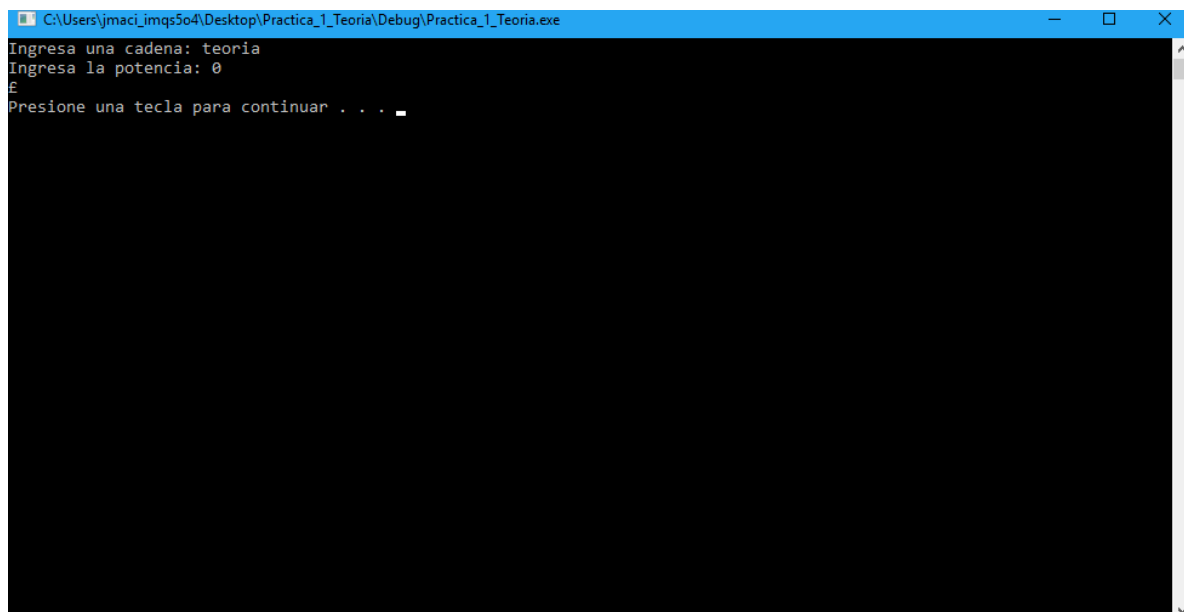
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Escribe la primer cadena: Teoria
Escribe la segunda cadena: Computacional
La concatenacion es: TeoriaComputacional
Presione una tecla para continuar . . .
```

La quinta opción muestra la potencia de la cadena, pero son 3 casos el primero es la potencia mayor a cero



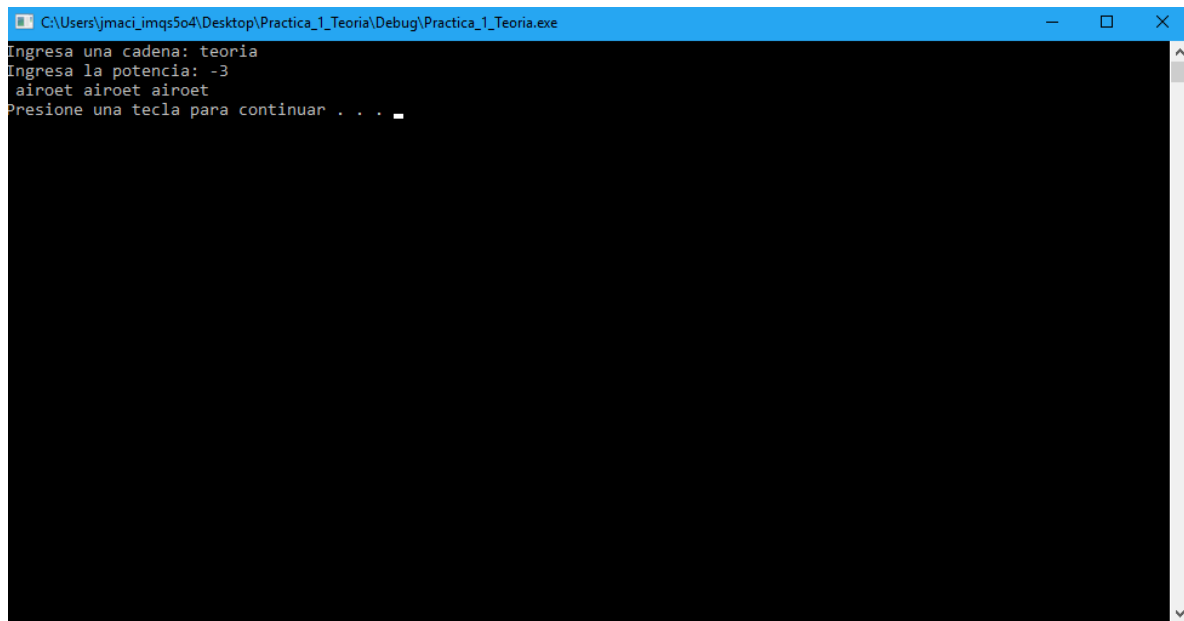
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoría\Debug\Practica_1_Teoría.exe
Ingresa una cadena: Teoria
Ingresa la potencia: 3
teoriaTeoriaTeoria
Presione una tecla para continuar . . .
```

Ahora la potencia igual a cero



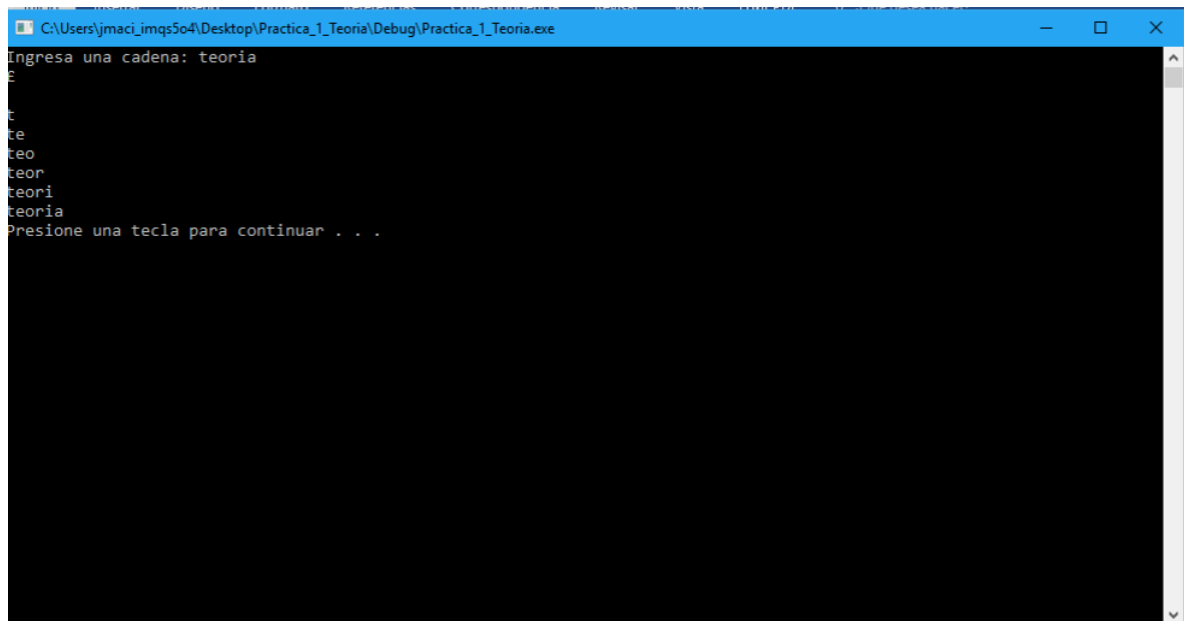
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoría\Debug\Practica_1_Teoría.exe
Ingresa una cadena: teoria
Ingresa la potencia: 0
Presione una tecla para continuar . . .
```

Por último, la potencia menor a cero



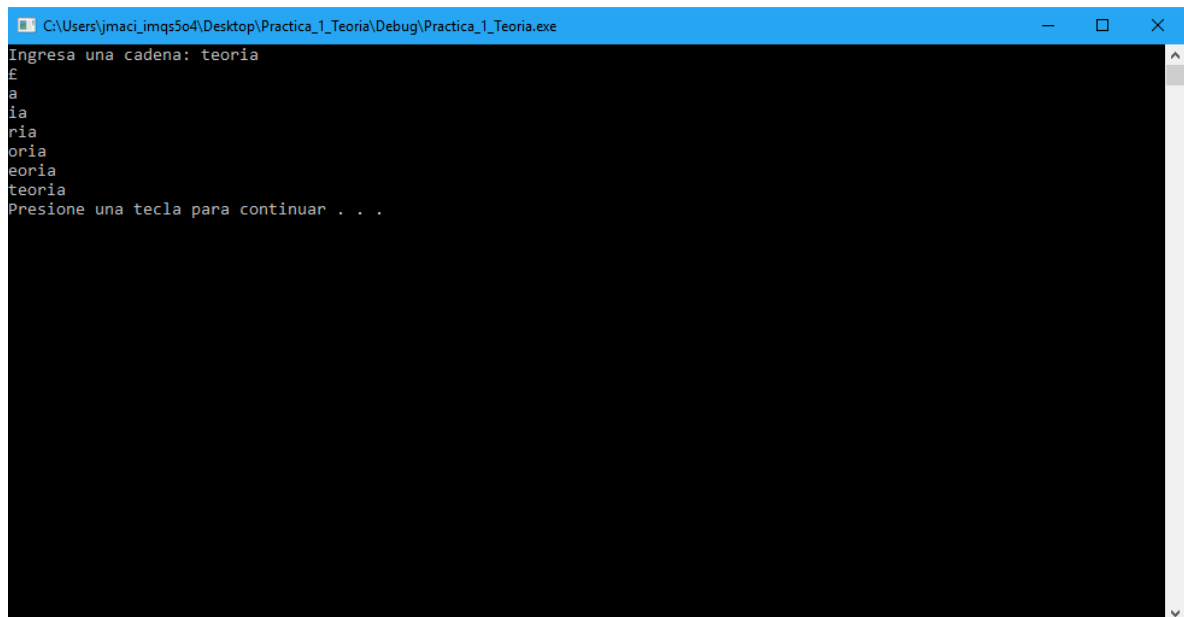
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Ingresa una cadena: teoria
Ingresa la potencia: -3
airoet airoet airoet
Presione una tecla para continuar . . .
```

La sexta opción del programa es el prefijo de la cadena



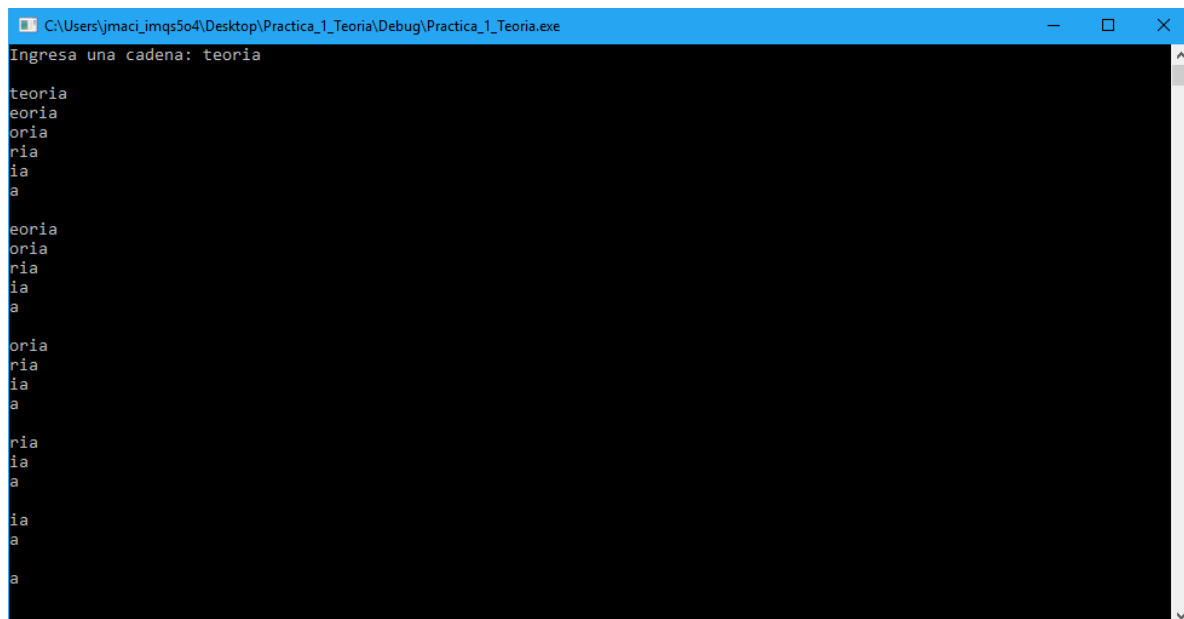
```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Ingresa una cadena: teoria
e
t
te
teo
teor
teori
teoria
Presione una tecla para continuar . . .
```

La séptima opción nos muestra el sufijo



```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Ingresa una cadena: teoria
t
e
o
r
i
a
Presione una tecla para continuar . . .
```

Y por último la opción de las subcadenas



```
C:\Users\jmaci_imqs5o4\Desktop\Practica_1_Teoria\Debug\Practica_1_Teoria.exe
Ingresa una cadena: teoria
te
eo
or
ri
ia
a
Presione una tecla para continuar . . .
```

Conclusión:

La práctica se me hizo muy sencilla, aunque admito que tuve alguno problemita con las subcadenas, pero después de muchos intentos al fin quedo, me agrado mucho hacer la implementación de las operaciones con cadenas ya tenía tiempo que no programaba en C y esta práctica me hizo recordar todo, gracias a esta práctica ya entiendo de mejor forma las operaciones de cadenas.

Bibliografía

John E. Hopcroft, R. M. (2007). *Teoria de autómatas, lenguajes y computación*. Madrid: PEARSON EDUCACION S.A.