

PRACTICA 11: PALÍNDROMOS

**ESTUDIANTE: EFRAIN ROBLES PULIDO** 

CODIGO: 221350095

NOMBRE DE LA MATERIA: SEMINARIO DE SOLUCION DE PROBLEMAS DE PROGRAMACION

SECCIÓN: D67

## Descripción:

## Objetivo de la practica

Hacer un programa para determinar si la frase ingresada por el usuario es un palíndromo, sino no lo es, debe solicitarse otra frase, el programa termina cuando la frase sea palíndroma.

### Fundamentación teórica



Un palíndromo es una palabra o frase cuyas letras están dispuestas de tal manera que resulta la misma leída de izquierda a derecha que de derecha a izquierda.

Habitualmente se entiende por palíndromo aquel que toma por unidad la letra, es decir, cuya última letra es la misma que la primera, la penúltima es la misma que la segunda, etc. Es el caso de palabras tales como reconocer o anilina. Sin embargo, también se puede

tomar como unidad la sílaba (por ejemplo, gato con toga, aunque en este caso podría ser calificado como anagrama), la palabra o incluso el renglón.

## Análisis del problema

Se deberá introducir la frase o palabra por analizar, y con la ayuda de las funciones de las cadenas podemos manipular nuestra cadena (frase o palabra) como copiarla a otra cadena a otra, o invertir la cadena y de comparar la cadena con otra. Y también se utilizará el ciclo for para poder quitar los espacios de la frase para tener una lectura correcta de la frase, además del ciclo do-while para que sea repetitivo en dado caso de que no sea palíndromo la frase o palabra por analizar.

### Datos de entrada y precondiciones

Frase o palabra por analizar

### Datos o elementos de salida

Mensaje de detección de palíndromos

#### Desarrollo:

## Procedimiento en lenguaje natural

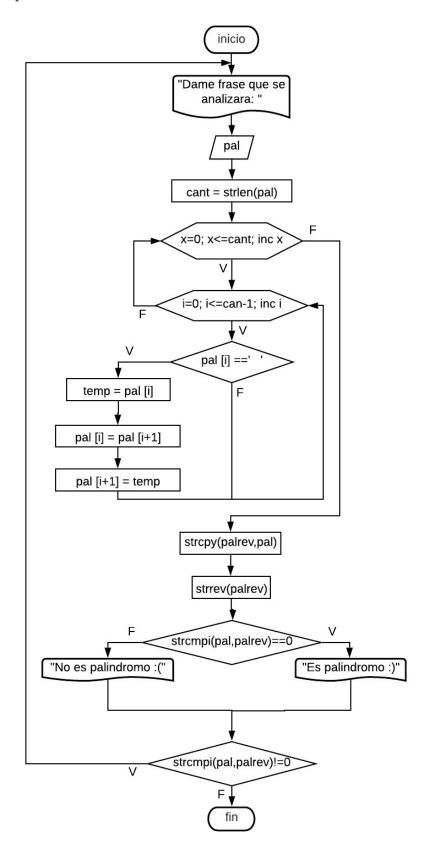
Se utilizo el ciclo do-while para que sea repetitivo el programa, haciendo las instrucciones primero, en donde empezara con la introducción de la frase o palabra por analizar con **gets**, después se obtiene la longitud de la cadena introducida con **strlen** para guardarla en la variable cant, después se utilizó 2 ciclos for anidado para reacomodar la cadena para mover los espacios al final de la cadena con la ayuda de la reasignación de variables para la cadena, después se utilizó las funciones **strcpy** para pasar lo de la cadena original a otra

cadena y **strrev** para invertir la cadena copia y así obtener la palabra invertida. Finalmente se analizará con if-else si es igual a 0 la función **strcmpi** de la cadena original y su invertida, si es verdadero mostrara que es palíndromo y sino dirá que no es palíndromo sin importar las mayúsculas, para después analizar en el do-while, si nos da diferente de 0 en la función **strcmpi** de la frase o palabra y su invertida, si cumple significara que no es palíndromo palabra o frase sin importar las mayúsculas volviendo e empezar el programa otra vez.

# **Algoritmo**

```
Pseudocódigo:
Inicio
        hacer
            inicio
                imprimir ("Dame frase que se analizara: ")
                obtener (pal)
                cant ← strlen(pal)
                desde (x \leftarrow 0; x<=cant; inc x)//Reacomodo de letras, manda los espacios al final
                inicio
                        desde (i \leftarrow 0; i<=cant-1; inc i)
                        inicio
                                si (pal[i]==' ')
                                inicio
                                        temp ← pal[i]
                                        pal[i] \leftarrow pal[i+1]
                                        pal[i+1] ← temp
                                fin
                        fin
                fin
                strcpy(palrev,pal)
                strrev(palrev)
                si (strcmpi(pal,palrev)==0)
                inicio
                        imprimir ("Es palindromo:)")
                fin
                sino
                        imprimir ("No es palindromo :(")
           fin
        mientras (strcmpi(pal,palrev)!=0)
```

Fin



## Código fuente del programa en lenguaje C

```
//Efrain Robles Pulido
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define TAM 100
int main(){
  char pal[TAM], palrev[TAM],temp;
  int cant,i,x;
  do{
     printf("Dame frase que se analizara: ");
     gets(pal);
     cant=strlen(pal);
     for(x=0;x<=cant;x++){//Reacomodo de letras, manda los espacios al final
       for(i=0;i\leq cant-1;i++)
          if(pal[i]==' '){
             temp=pal[i];
             pal[i]=pal[i+1];
             pal[i+1]=temp;
          }
       }
     strcpy(palrev,pal);
     strrev(palrev);
     if(strcmpi(pal,palrev)==0){
       printf("\n\tEs palindromo :)\n");
     }
     else{
       printf("\n\tNo es palindromo :(\n");
     }
     printf("\n");
  }while(strcmpi(pal,palrev)!=0);
}
```

#### Resultados obtenidos:

```
//Efrain Robles Pulido
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define TAM 100
int main() {//Palindromo con funciones
    char pal[TAM], palrev[TAM], temp;
    int cant, i, x;
    do
        printf ("Dame frase que se analizara: ");
        gets(pal);
        cant=strlen(pal);
        for (x=0; x<=cant; x++) (//Reacomodo de letras, manda los espacios al final
            for (i=0; i <= cant-1; i++) {
                if (pal[i] == ' ') {
                     temp=pal[i];
                     pal[i]=pal[i+1];
                    pal[i+1]=temp;
                }
           1
        strcpy(palrev,pal);
        strrev(palrev);
        if (strcmpi (pal, palrev) == 0) {
            printf("\n\tEs palindromo :)\n");
        else
            printf("\n\tNo es palindromo : (\n");
        printf("\n");
    }while(strcmpi(pal, palrev)!=0);
}
          Dame frase que se analizara: Dejan mucha tarea en la escuela
                  No es palindromo :(
          Dame frase que se analizara: Yo solo se que no se nada
                  No es palindromo :(
          Dame frase que se analizara: Ana la loca saco la lana
                  Es palindromo :)
          Process returned 0 (0x0) execution time : 42.256 s
          Press any key to continue.
```

## Conclusión

En esta práctica fue interesante utilizar las funciones que nos ofrece la librería de cadenas (string), los ciclos for para poder hacer corrimientos en la cadena para poder cumplir el objetivo de la práctica. Se tuvo que juntar toda la cadena con un arreglo de for anidados y una reasignación de variables en la cadena, para después hacer una copia en otra cadena para poder invertirlo y poder analizar la cadena original con la invertida y si resultaba un valor diferente a 0, sin importar las mayúsculas de la frase o palabra, el programa se repetiría hasta que sean iguales las cadenas (la original y la invertida).