PALINDROMO

BRIGADA 6

Análisis, describiendo el problema e identificando los datos de entrada y de salida "palíndromo".

Para la realización de nuestro programa sobre palíndromo, debemos adentrarnos más sobre el concepto de este sistema y sobre el significado de la palabra "Palíndromo", la cual hace referencia a una palabra o frase que se lee igual de derecha a izquierda y viceversa, tales como:

Adán – Nada

Amor – Roma

Animal – Lámina

Entre otros más…

Para este proyecto nuestro programa debe validar si una cadena es un palíndromo. Sin tomar en cuenta diferencias entre mayúsculas y minúsculas ni los espacios.

El desarrollo de nuestro código fue el siguiente:

Como punto de inicio sobre nuestros datos usamos 4 librerías:

- ***Stdio.h:*** El archivo de cabecera que contiene las definiciones de las macros, las constantes, las declaraciones de funciones de la biblioteca estándar del lenguaje de programación C para hacer operaciones, estándar, de entrada y salida, así como la definición de tipos necesarias para dichas operaciones.

- ***String.h:*** Es un archivo de la Biblioteca estándar del lenguaje de programación C que contiene la definición de macros, constantes, funciones y tipos y algunas operaciones de manipulación de memoria.

- ***Stdlib.h:*** Es el archivo de cabecera de la biblioteca estándar de propósito general del lenguaje de programación C. Contiene los prototipos de funciones de C para gestión de memoria dinámica, control de procesos y otras.

- ***Ctype.h:*** Es un archivo de cabecera de la biblioteca estándar del lenguaje de programación C diseñado para operaciones básicas con caracteres. Contiene los prototipos de las funciones y macros para clasificar caracteres.

Por consiguiente, se empezó a declarar **int main** el cual es nuestro objeto de inicio sobre la entrada de datos que vamos a utilizar y generar para nuestro código, iniciando con **char**, el cual es usualmente utilizado como parte de cadenas de caracteres, como el que haremos en nuestro programa junto con **malloc**, función que asigna memoria dinámica, es decir, solicita un nuevo bloque de memoria libre de la asignada al programa, el cual va fijo con la longitud que este puede alcanzar *"100"* haciendo referencia a que la extensión máxima de las frases o palabras que se ingresen para el funcionamiento de nuestro programa no excedan este límite, permitiendo su total funcionamiento.

Después, tenemos la función **printf** la cual nos mostrará un cuadro de texto en el programa, siendo este "Escribe la palabra", dando pauta a que el usuario ingrese la frase o palabra para poder analizarla, y finalmente decir si cumple con el objetivo el cual es diferenciar entre si es palíndromo o no.

Gets, se encarga de leer y almacenar una cadena de caracteres introducida por el usuario, la cual es guardada hasta que haya un salto de línea ("\ n") en donde cabe mencionar que el salto de línea no se guarda, es capaz de guardar cualquier tipo de carácter incluidos espacios en blanco y no tiene ningún limitante en cuanto al tamaño de la cadena de carácter que se introduzca, a menos que el usuario designe ese límite.

Int. Declara más variables junto con su valor, para posteriormente poder evaluar la frase y la longitud que esta tiene. **Strlen** sirve como función del código el cual devuelve la longitud de una cadena de texto, regresándola como un número entero (número de letras almacenadas, sin contar el carácter nulo final).

Char x " ". Esta cadena se diferencia para poder escribir la frase o palabra, pero de manera inversa, ayudando al programa para verificar si los valores que el usuario introduce cumplen con el objetivo de ser un palíndromo.

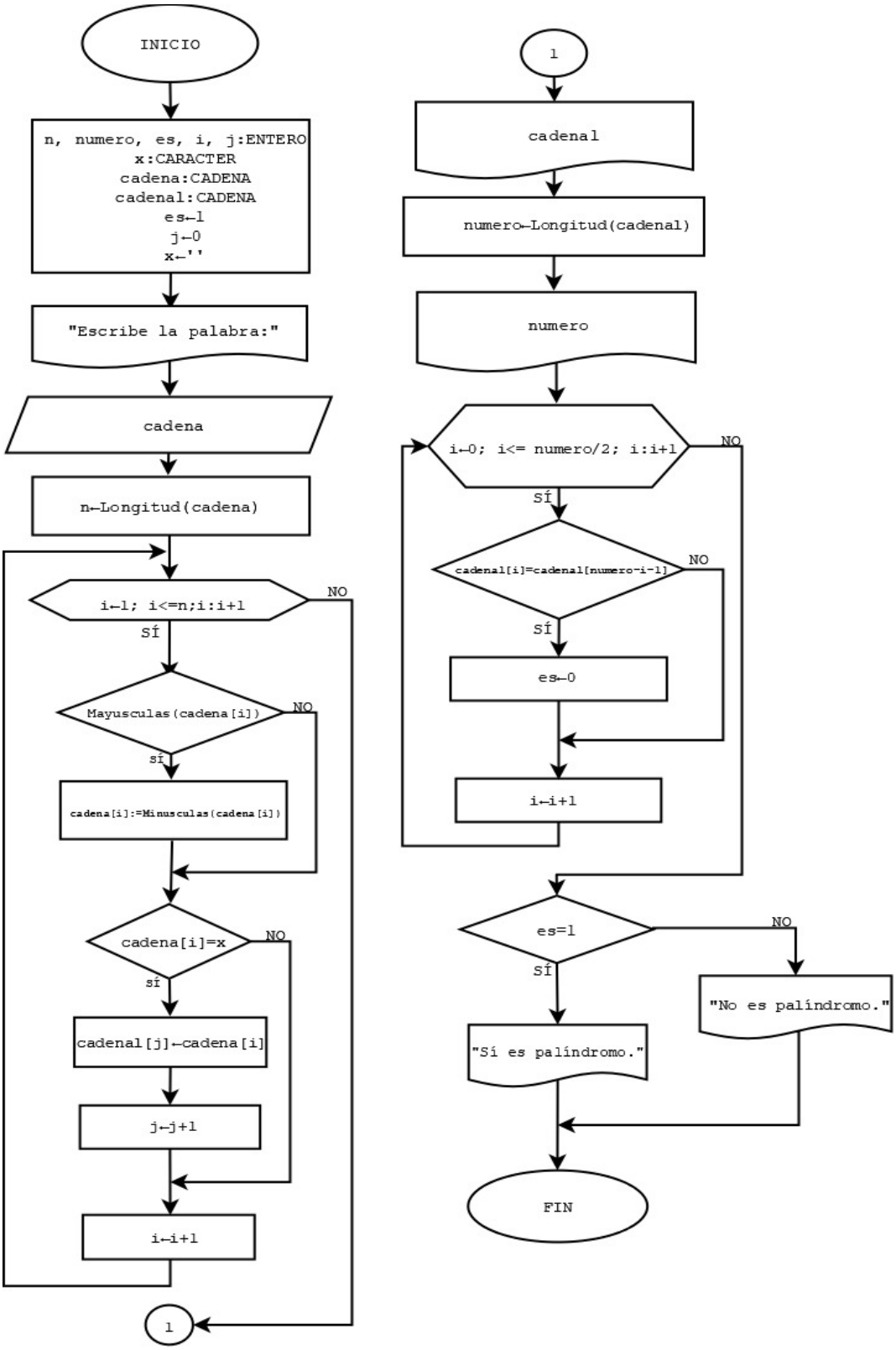
Para el paso siguiente la variable n deberá sr igual a la función **strlen** involucrando a la cadena que se introduce, iniciando el proceso para poder revertir los parámetros a diferenciar en el programa, en donde se toman los valores numéricos de letras desde 1 para efectuar la función hacer, haciendo uso de la cadena con la frase utilizada en dónde la condición **If** seguida de la función **tolower** permite la entrada tanto de letras mayúsculas como minúsculas para no tener ningún tipo de error durante el proceso de lectura, después de terminar estas acciones para almacenar la información de nuestra palabra en la memoria que se toma en cuenta sobre el programa, de misma manera después de verificar los datos guardados en la frase y la condición, hace la enumeración de los caracteres que se tienen e imprime en pantalla la cadena que se tiene, aquí es muy importante recalcar que se activa el salto de línea mencionado anteriormente (\n), activando a su vez la función **gets**, con el número de letras sumándolas y revirtiendo su sentido para identificar si efectivamente cumple las condiciones, por lo que el siguiente y último condicional **if** hace que sí la palabra cumple lo dicho con anterioridad, arroje el mensaje con la función **printf** "Si es palíndromo" o en caso contrario **else** "No es palíndromo", después de arrojar estas dos opciones de datos, la función **free** libera la memoria utilizada por la frase o palabras ingresadas en los primeros pasos, para que el usuario continúe probando con diferentes enunciados para comprender su funcionamiento total y efectivo.

Pseudocodigo

```
INICIO
  n, numero, es, i, j:ENTERO
  x:CARACTER
  cadena:CADENA
  cadenal:CADENA
  es:=1
  j:=0
  x:="
  ESCRIBIR "Escribe la palabra"
  LEER cadena
  n:=Longitud(cadena)
  PARA i:=1 HASTA i<=n HACER
     SI Mayusculas(cadena[i]) ENTONCES
          cadena[i]:=Minusculas(cadena[i])
     FIN SI
     SI cadena[i]=x ENTONCES
         cadenal[j]:=cadena[i]
         j:=j+1
     FIN SI
     i:=i+1
  FIN PARA
  ESCRIBIR cadenal
  numero:=Longitud(cadenal)
  ESCRIBIR numero
  PARA i:=0 HASTA i<= numero/2 HACER
     SI cadenal[i]=cadenal[numero-i-1] ENTONCES
        es:=0
     FIN SI
```

```
    i:=i+1
  FIN PARA
  SI es=1 ENTONCES
    ESCRIBIR "Si es palíndromo"
  FIN SI
  DE LO CONTRARIO
    ESCRIBIR "No es palíndromo"
  FIN DE LO CONTRARIO
FIN
```

Diagrama de flujo



Programa en C

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

int main(){
  char* cadena = (char*)malloc(100);
  char* cadena1 = (char*)malloc(100);
  printf("Escribe la cadena\n");
  gets(cadena);
  int n, numero, es, i, j;
  es = 1;
  j = 0;
  char x = ' ';
  n=strlen(cadena);
  for(i=0;i<=n;i++){
    if (isupper(cadena[i]))
      cadena[i] = tolower(cadena[i]);
    if(cadena[i] != x){
      cadena1[j] = cadena[i];
      j++;
    }
  }
  printf("%s\n", cadena1);
  numero = strlen(cadena1);
  printf("%d\n", numero);
  for(i=0;i<=numero/2;i++){
    if(cadena1[i] != cadena1[numero-i-1])
```

```
    es = 0;

  }
  if(es)
    printf("Si es palindromo\n");
  else
    printf("No es palindromo\n");

  free(cadena);
  free(cadena1);

}
```

PRUEBA DE ESCRITORIO

```
Escribe la palabra
anita lava la tina
anitalavalatina
15
Si es palindromo

Process returned 0 (0x0)   execution time : 92.385 s
Press any key to continue.
```

```
Escribe la palabra
A Mafalda dad la fama
amafaldadadlafama
17
Si es palindromo

Process returned 0 (0x0)   execution time : 2.354 s
Press any key to continue.
```

TEST

```
/*
char cadena[] = "";
verificarPalindromo(cadena), 0
char cadena[] = "anita lava la tina";
verificarPalindromo(cadena), 1
char cadena[] = "A Mafalda dad la fama";
verificarPalindromo(cadena), 1
*/

#include "..\index.c"
int main()
{
    char cadena[] = "";
    char cadena[] = "anita lava la tina";
    char cadena[] = "A Mafalda dad la fama";

    if(verificarPalindromo(cadena), 0)
    {
        printf("Si es palindromo\n");
    }
    else
    {
        printf("No es palindromo\n");
    }

    if (verificarPalindromo(cadena), 1)
```

```
    {
        printf("Si es palindromo\n");
    }
    else
    {
        printf("No es palindromo\n");
    }


    if (verificarPalindromo(cadena), 1)
    {
        printf("Si es palindromo\n");
    }
    else
    {
        printf("No es palindromo\n");
    }
}
```