Nombre de la práctica	CADENAS Y FUNCIONES			No.	3
Asignatura:	MÉTODOS NUMÉRICOS	Carrera:	SISTEMAS COMPUTACIONALES	Duración de la práctica (Hrs)	5 horas

NOMBRE DEL ALUMNO: DALILA SOTO HERNADEZ

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

- Casa

III. Material empleado:

- DEV C++

1.-

Con ayuda de **printf()**, y de con **scanf()** guardamos el valor que el usuario nos de através del teclado, este valor se asigna al especificador de acceso para despues guardar dicho valor en la variable correspondiente, en este caso, cad. Al final nuevamente con printf() mostramos lo que el usuario nos ingresó, e igual se asigna al especificador de conversión.

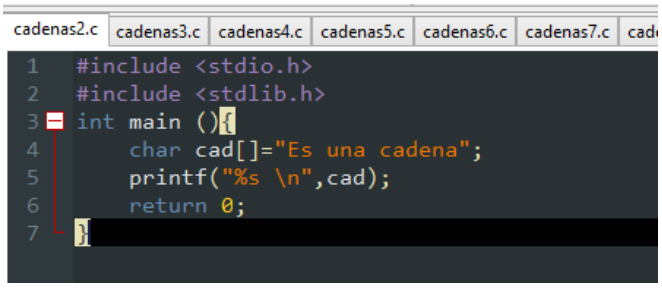
Declaración de una cadena, consta de su tipo de dato (char), el nombre (cad) y su tamaño (10), al igual que una variable llamada i de tipo entero, designada para el ciclo.

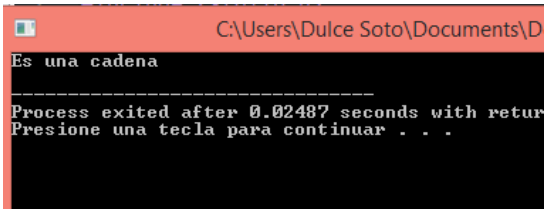
cadenas.c | cadenas2.c | cadenas3.c | cadenas4.c | cadenas5.c | cadenas6.c | cadenas7.c | cadenas

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int main (){
4      char cad[10];
5      int i;
6      printf("Introduce una palabra: \n");
7      scanf("%s",cad);
8      printf("%s",cad);
9      return 0;
10 }
```

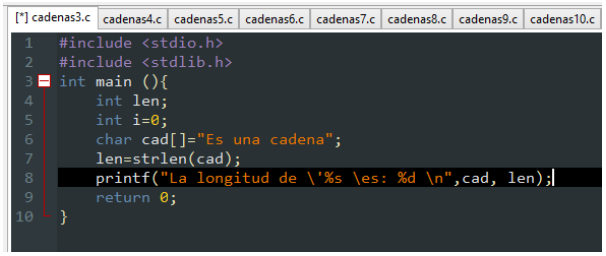
```
C:\Users\Dulce Soto\Documents\DevC++\cad
Introduce una palabra:
CADENAS
CADENAS
--------------------------------
Process exited after 4.748 seconds with return value 0
Presione una tecla para continuar . . .
```

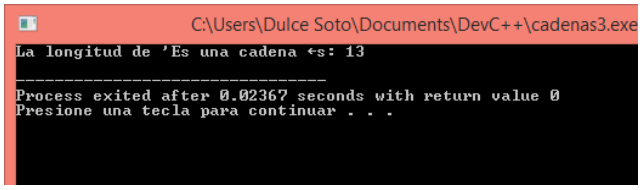
2.-
Con **printf()** mostramos el valor de la cadena en pantalla, el valor se le asigna al especificador de conversión que espera un tipo de dato char.
El especficador de conversion es el siguiente: **%s**





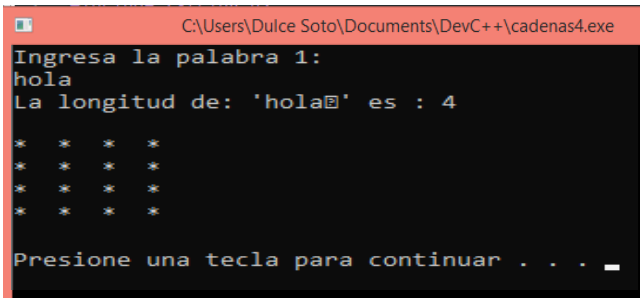
3.-
Con la función **strlen** obtenemos la longitud de cadena, es decir, cuantos valores tiene, y el resultado se guarda en la variable **len**, para mostrar la cadena y su longitud en pantalla utilizamos **printf().**





4.-

El ciclo anidado nos va a ayudar para imprimir un cuadrado con (*) de n*n dependiendo de la longitud de la palabra, ambos ciclos los inicializamos en cero, y la condición es que sea menor al tamaño del arreglo por eso están en cero al iniciar, y al ir incremento va a imprimir un *, cuando termine el primer ciclo imprime un salto de línea.

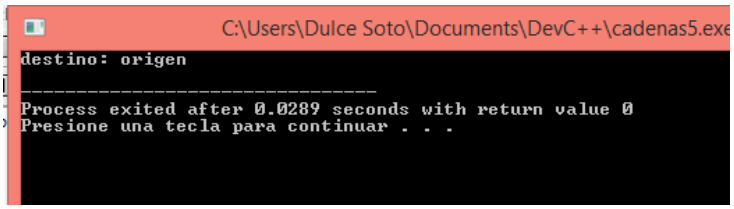


5.-

Declaramos los arreglos antes, después y nuevo, los tres de tipo char.

Con **puts()** mandamos un mensaje en pantalla con la instrucción de ingresa una palabra mientras que con scanf() podemos obtener el dato que el usuario nos vaya a ingresar por teclado, este se asigna al especificador de conversión en la variable designada para cada uno.

Con **strcpy,** copiamos el valor de la cadena origen a la cadena destino lo que se esta haciendo es transferir lo que tiene almacenado una cadena a la otra pero una cadena no puede almacenar lo de dos valores así para eso creamos la 3 cadena que guarda el valor de una mientras se pasa el valor de una cadena a otra.



6.-

Declaramos una variable llamada len de tipo entero, y dos cadenas de tipo char, una llamada destino con el contenido “brisas” y la segunda con un tamaño de 11 caracteres, que ya tiene dentro de la palabra para.

Con la función **strcat** concatenamos brisas a para, y el nuevo contenido de destino es parabrisas

```c
#include <stdio.h>
#include <string.h>
int main (){
    char antes[]="antes";
    char despues[]="despues";
    char nuevo[]="nueva cadena";
    puts("Ingresa una palabra: ");
    scanf("%s", &antes);
    puts("Ingresa otra palabra: ");
    scanf("%s", &despues);
    strcpy(nuevo, antes);
    strcpy(antes, despues);
    strcpy(despues, antes);
    strcpy(antes, despues);

    printf("despues: %s\n antes: %s", despues,nuevo);
    return 0;
}
```

```
C:\Users\Dulce Soto\Documents\DevC++\cadenas
Ingresa una palabra:
hola
Ingresa otra palabra:
mundo
despues: mundo
 antes: hola
--------------------------------
Process exited after 4.297 seconds with return value 0
Presione una tecla para continuar . . .
```

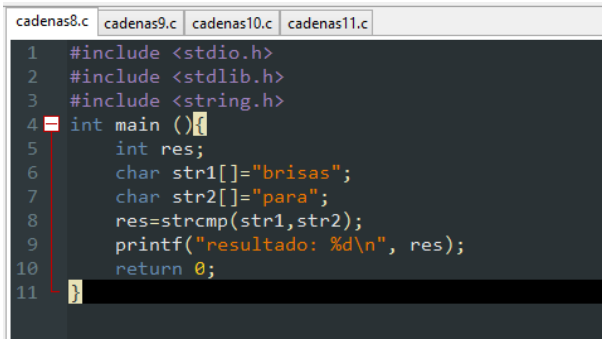
7.-

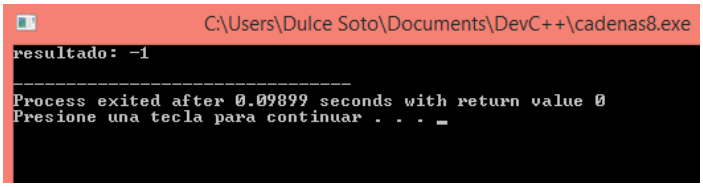
Al utilizar la función **strcmp** comparamos “para” y “brisas”, y el resultado de esa comparación se guarda en la variable res, que después se imprime en pantalla, el valor se guarda en el especificador **%d** de tipo decimal

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (){
    int len;
    char origen[]="brisas";
    char destino[11]="para";
    strcat(destino,origen);
    printf("destino: %s", destino);
    printf("\n");
    return 0;
}
```

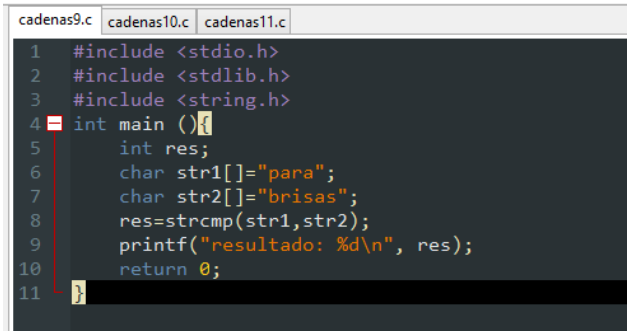
```
C:\Users\Dulce Soto\Documents\DevC++\cadenas7.exe
destino: parabrisas

--------------------------------
Process exited after 0.03008 seconds with return value 0
Presione una tecla para continuar . . .
```

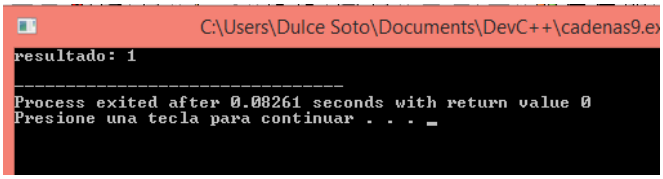
8.-
Declaramos el dato de tipo entero **res** y las dos cadenas una con el nombre de **str1** y la otra **str2** una con el valor asignado de brisas y la otra con para asignado, En este caso, brisas es mayor que para, entonces devuelve un numero negativo.





9.-
En caso contrario como se puede ver en este resultado en donde la palabra para es menor que brisas entonces devuelve un resultado positivo.





10.-

Declaramos las variable z, x y res de tipo entero, también las declaramos las cadenas cad1, cad2, mayor e igual, como son cadenas son char, cad1 y cad2, están iniciadas como una cadena vacía, esto es porque aun no sabemos que tamaño tendrán , ya que se ingresaran por teclado, y mayor e igual tienen un mensaje después de la comparación que será el que se muestre después de la ejecución.

con **printf()** mandamos un mensaje en pantalla, y con gets() obtendremos los valores, segun la cadena que el usuario elija por teclado, el valor se va a guardar en la variable cad1 y cad2 respectivamente, luego con **strcmp()** se comparan las 2 cadenas y el resultado se guarda en res y finalmente se imprime.

cadenas10.c | cadenas11.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (){
    int z, x;
    int res;
    char cad1[]="     ";
    char cad2[]="     ";
    char mayor[]="la palabra 1 es menor";
    char igual[]="las palabras son iguales";

    printf("Ingresa la palabra 1: ");
    gets(cad1);
    printf("Ingresa la palabra 2: ");
    gets(cad2);

    res=strcmp(cad1, cad2);
    printf("%d\n", res);

    printf("%s : %d", (z=res ? mayor: igual),(x=res ? 1:0));
    printf("\n");

    return 0;
}
```

```
C:\Users\Dulce Soto\Documents\DevC++\cadenas10.exe
Ingresa la palabra 1: para
Ingresa la palabra 2: brisas
1
la palabra 1 es menor : 1

--------------------------------
Process exited after 4.141 seconds with return value 0
Presione una tecla para continuar . . .
```

```
C:\Users\Dulce Soto\Documents\DevC++\cadenas10.exe
Ingresa la palabra 1: brisas
Ingresa la palabra 2: brisas
0
las palabras son iguales : 0

--------------------------------
Process exited after 13.49 seconds with return value 0
Presione una tecla para continuar . . .
```

11.-

Declaramos las cadenas copia y cad1 inicializadas vacías de tipo char, las variables i, j y len de tipo entero, en su caso con es un contador, y en len se guardará la longitud de cad, y j e i son identificadores del ciclo.

Con ayuda de la función **puts()** mandamos un mensaje , con **gets()** obtenemos por teclado la cadena que el usuario ingrese, **strcpy()** nos sirve para copiar el mismo contenido a otro arreglo en este caso de cad1 a copia y con **strlen()** obtenemos la longitud de la cadena cad1, la cual le asignaremos este valor+1 como tamaño de la cadena y repet de tipo entero.

En el ciclo anidado, empezamos a recorrer las cadenas desde el final y vamos comparando posición por posición si alguna de las letras se encuentra más de una el contador va a incrementar, pero si la letra es igual entonces se le va a asignar un valor de 0 para que ya no se imprima doble vez, después el valor que haya tomado cont se le asignara como valor a la posición indicada de la cadena repet, y así sucesivamente hasta completar el ciclo.

[*] cadenas11.c

```c
int main (){
    char cad1[]=" ";
    char copia[]=" ";
    int i, j, len, con=0, cont=0;

    puts("Ingresa una palabra: ");
    gets(cad1);
    len=strlen(cad1);
    strcpy(copia,cad1);
    int repet[len+1];
    for(i=len;i>=0;i--){
        for(j=len;j>=0;j--){
            if(cad1[i]==copia[j]){
                con+=1;
            }
            if(cad1[i]==copia[j] && i>j){
                con=0;
            }
        }
        repet[i]=con;
        con=0;
    }
    for(i=0;i<len;i++){
        if(repet[i]>0){
            printf("%c : %d\n",copia[i], repet[i]);
        }
    }
    printf("\n");
    return 0;
```

```
C:\Users\Dulce Soto\Documents\DevC++\cadenas11.ex
Ingresa una palabra:
palabra
p : 1
a : 3
l : 1
b : 1
r : 1

--------------------------------
Process exited after 13.38 seconds with return value 0
Presione una tecla para continuar . . .
```

12.-

Declaramos una cadena con el nombre cad para caracteres con un tamaño de 10 y la variable i de tipo entero.

Printf() hace referencia a una impresión en pantalla con la instrucción de ingresar una palabra, esta con ayuda de la función **gets(),** va a permitir ingresarla por teclado y los caracteres los va a ir guardando en el arreglo cad. después se le asigna ese arreglo al especificador **%s** de conversión de caracteres y los muestra en pantalla.

funciones.c

```c
#include <stdio.h>
#include <stdlib.h>
int main (){
    char cad[10];
    int i;
    printf("Ingresa una palabra: \n");
    gets(cad);
    printf("\n%s", cad);
    printf("\n");
    return 0;
}
```

```
C:\Users\Dulce Soto\Documents\DevC++\funciones.
Ingresa una palabra:
hola mundo

hola mundo

--------------------------------
Process exited after 4.324 seconds with return value 0
Presione una tecla para continuar . . .
```

13.-

Declaramos las variables x, y y mayor de tipo entero.

El método máximo con tipo de retorno entero, contiene dos parámetros que son valores que se tomaran a partir de la llamada por el método main, estos valores tienen que ser obligatoriamente, asignados, dentro de este método tenemos una declaración de variable llamada mayor de tipo entero, luego se compara si x es mayor a y, si es así, el valor de x se guarda en mayor y será el valor que se devuelva, de no se así se ejecuta lo contrario.

Mandamos una impresión por pantalla para que el usuario ingrese el número 1 y 2 a comparar, esto se hace por medio de **printf()**, mediante **scanf()** nosotros podemos obtener el valor que el usuario nos ha ingresado por teclado, y el valor se guarda en la variable correspondiente, en la variable mayor se va a guardar el valor que el método máximo va a devolver, pasándo como parámetros los numeros 1 y2, que se encuentran en la variable x y y, después solo imprimimos el valor arrojado por el método máximo.

[*] funciones.c | funciones2.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (){
    int x, y;
    int mayor;
    printf("Ingrese el numero 1: ");
    scanf("%d",&x);
    printf("Ingrese el numero 2: ");
    scanf("%d",&y);
    mayor= maximo(x,y);
    return 0;
}
int maximo(int x, int y){
    int mayor;
    if(x>y){
        mayor=x;
    }else{
        mayor=y;
        printf("el numero mayor es: %d",mayor);
    }
    return mayor;
}
```

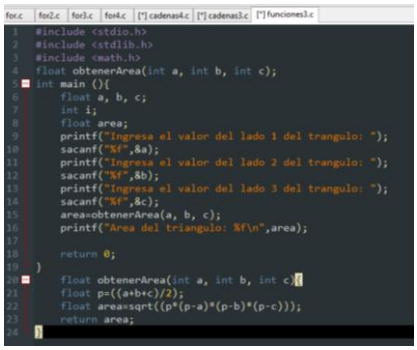
```
C:\Users\Dulce Soto\Documents\DevC++\funciones2.exe
Ingrese el numero 1: 50
Ingrese el numero 2: 90
el numero mayor es: 90
--------------------------------
Process exited after 3.486 seconds with return value 0
Presione una tecla para continuar . . . _
```

14.-

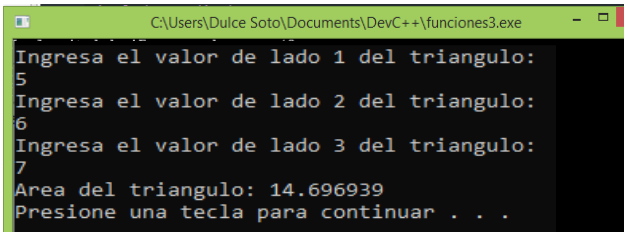
Con la función **printf()** podemos mostrar un mensaje en pantalla, mientras que con **scanf()** obtenemos el valor introducido por teclado por parte del usuario, el valor se asigna a la variable correspondiente.

En la variable área, se guarda el valor que retorna el método obtenerArea, asi al mandar llamar la le pasa como parameros el valor de a, b y c, y después se imprime.

El método obtenerArea con tipo de retorno float, contiene 3 parámetros, que son los valores de los lados del triángulo, valores que serán proporcionados por el método main, al momento de hacer la llamada, dentro se declara a p de tipo float, que guarda el resultado de la división de la suma de todos sus lados entre 2.



```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
float obtenerArea(int a, int b, int c);
int main (){
    float a, b, c;
    int i;
    float area;
    printf("Ingresa el valor del lado 1 del trangulo: ");
    sacanf("%f",&a);
    printf("Ingresa el valor del lado 2 del trangulo: ");
    sacanf("%f",&b);
    printf("Ingresa el valor del lado 3 del trangulo: ");
    sacanf("%f",&c);
    area=obtenerArea(a, b, c);
    printf("Area del triangulo: %f\n",area);

    return 0;
}
    float obtenerArea(int a, int b, int c){
    float p=((a+b+c)/2);
    float area=sqrt((p*(p-a)*(p-b)*(p-c)));
    return area;
}
```



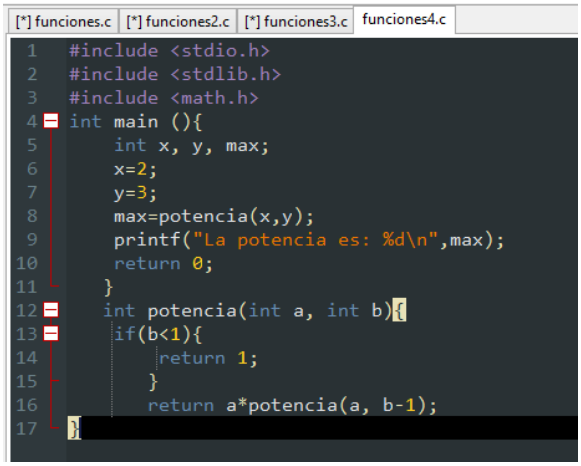
```
C:\Users\Dulce Soto\Documents\DevC++\funciones3.exe
Ingresa el valor de lado 1 del triangulo:
5
Ingresa el valor de lado 2 del triangulo:
6
Ingresa el valor de lado 3 del triangulo:
7
Area del triangulo: 14.696939
Presione una tecla para continuar . . .
```

15.-

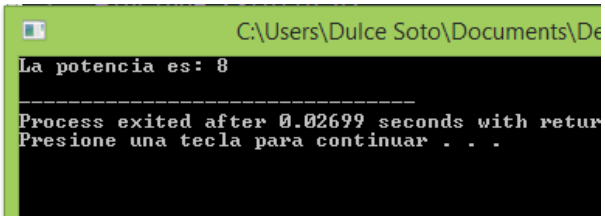
Declaramos variables x, y y max, asi mismo hay una asignación de un valor a x y y,

Dentro del método main, se manda a llamar al método potencia que es recursivo, aquí se devuelve un valor final, que se va a guardar en le variable max, después se imprime esta variable con ayuda del especificador de conversión **%d**

El método potencia es un método recursivo, es decir, que su función es parecida a la de un ciclo, pero debe terminar alguna vez, es por ello que se auxilia de una condición, su función es realizar un proceso y volverse llamar a si mismo. En este ejercicio nos pide la potencia, nos dice que x vale 2 y y vale 3, entonces estos valores se le pasan al método potencia.



```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main (){
    int x, y, max;
    x=2;
    y=3;
    max=potencia(x,y);
    printf("La potencia es: %d\n",max);
    return 0;
   }
   int potencia(int a, int b){
    if(b<1){
        return 1;
       }
       return a*potencia(a, b-1);
}
```



```
C:\Users\Dulce Soto\Documents\De
La potencia es: 8

--------------------------------
Process exited after 0.02699 seconds with retur
Presione una tecla para continuar . . .
```