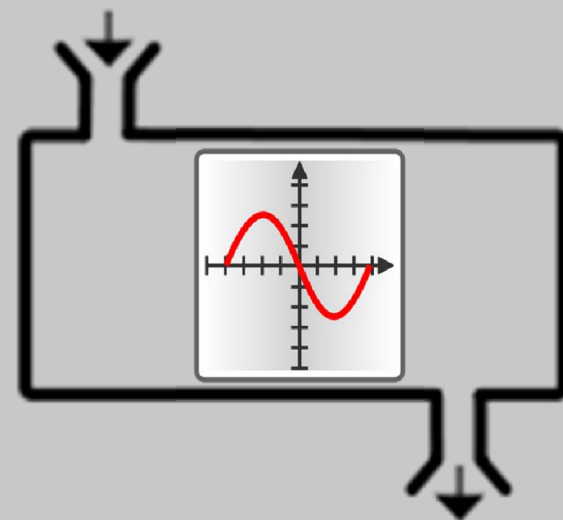


INPUT x



OUTPUT $f(x)$

FUNCIONES

Definición de función matemática

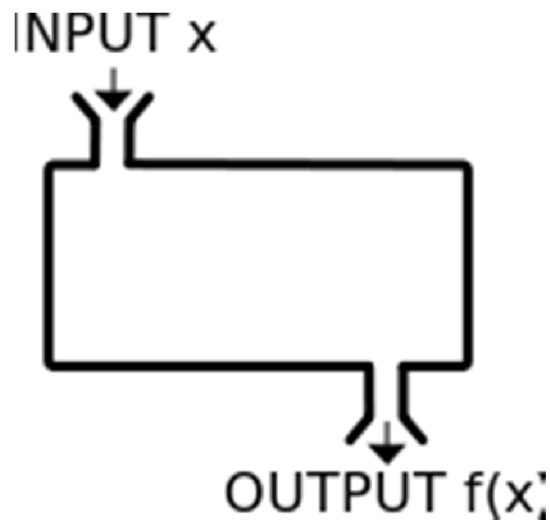
2

- Matemáticamente una función es una **operación** que toma uno o más valores llamados **argumentos** y produce un valor llamado **resultado**.

Definición de función

3

- Una función es un bloque de código reconocido por un identificador que realiza un trabajo específico.
- Su propósito es dividir los programas en módulos manejables separados (divide y vencerás).



Ventajas

4

1. Facilita el diseño descendente.
2. Los procedimientos dentro de ellas se pueden ejecutar varias veces.
3. Facilita la división de tareas.
4. Se pueden probar individualmente
5. Con funciones apropiadamente diseñadas, es posible ignorar como se realiza una tarea, sabiendo qué es lo que hacen.

Modo de uso

5

1. Funciones diseñadas para realizar operaciones a partir de sus argumentos y devolver un valor basado en sus cálculos.
2. Funciones que no reciben argumentos, realizan un proceso y devuelven un valor .
3. Funciones que no tienen argumentos ni valor de retorno explícito, operan sobre el entorno de variables globales o atributos del sistema operativo.

6

Ejemplo de funciones

```
int suma (int a, int b){  
    int c;  
    c = a + b;  
    return c;  
}
```

```
float promedio(int a, int b, int c){  
    int sum = a + b + c;  
    float prom = sum/3;  
    return prom;  
}
```

Declaración de funciones

7

- El formato para la declaración de funciones es:

```
tipo nombre_funcion(lista de parámetros) {  
    cuerpo de la función  
}
```

- **tipo** : especifica el tipo de valor que devuelve la función. Si no se especifica tipo, el compilador asume que es entero (int).

Declaración de funciones (2)

8

```
tipo nombre_funcion(lista de parámetros) {  
    cuerpo de la función  
}
```

- **lista de parámetros** : es la lista de nombres de variables separados por comas con sus tipos asociados que reciben los valores de los argumentos actuales de la llamada a la función.
- Entre llaves se encuentra el cuerpo de la función.

Declaración de funciones (3)

9

- Forza la salida inmediata de la función en que se encuentra.
- Una función puede retornar valor sólo cuando el tipo de retorno no es **void**.
- Devuelve un valor a la función que realizó la llamada.
- **return (expresion);**



Declaración de funciones (4)

10

- ❑ Tradicionalmente en C se declaran como prototipos al inicio del programa. Después se declara la función main, y después se hace la declaración formal de las funciones.
- ❑ También pueden declararse las funciones al inicio del programa y después declarar la función main sin declarar prototipo.

Ejemplo de funciones

```
int suma (int a, int b){  
    int c;  
    return int  
    c = a + b;  
}
```

```
float promedio(int a, int b, int c){  
    int sum = a + b + c;  
    float prom = sum/3;  
  
}
```

Llamadas a funciones

12

- Para llamar a una función se especifica su nombre y la lista de argumentos sin poner el tipo de dato.

```
nombre_funcion (var1, var2, ... varN)
```

- En una llamada habrá un argumento real por cada argumento formal, respetando el orden de declaración.
 - ▣ argumento formal: Los que aparecen en la definición de la función.
 - ▣ argumento real: Los que se pasan en la llamada a la función.

Paso de parámetros por valor

13

- Se hace una copia del valor del argumento en el parámetro formal.
- La función opera internamente con estos últimos.
- Los parámetros formales se crean al entrar a la función y se destruyen al salir de ella, cualquier cambio realizado por la función en los parámetros formales no tienen ningún efecto sobre los argumentos.

Ejercicio con funciones

```
float promedio (float a, float b);
```

```
int main(){
```

```
float a = 5, b=10 , prom;
```

```
prom = promedio (a, b);
```

```
printf ("El promedio es: %2.1f\n", prom);
```

```
system ("Pause");
```

```
return 0;
```

}

```
float promedio (float a, float b){
```

```
float prom;
```

a = a + 3;

b = b + 3;

```
prom = (a + b)/2;
```

```
return prom;
```

}

[illegible]

Ejercicio con funciones

```
float promedio (float a, float b);
```

```
int main(){
```

```
float a = 5, b=10 , prom;
```

```
prom = promedio (a, b);
```

```
printf ("El promedio es: %2.1f\n", prom);
```

```
system ("Pause");
```

```
return 0;
```

}

```
float promedio (float a, float b){
```

```
float prom;
```

a = a + 3;

```
b = b + 3;
```

```
prom = (a + b)/2;
```


```
return prom;
```

}

[illegible]

Ejercicio con funciones

16



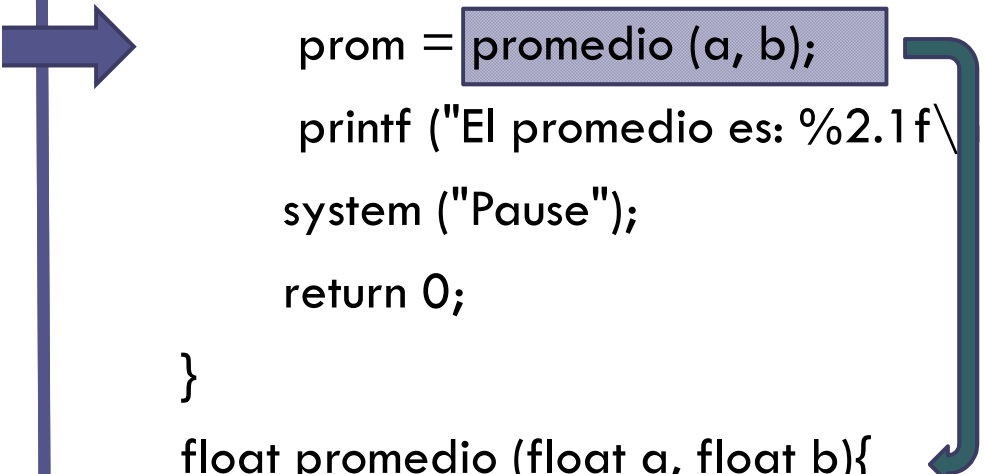
```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	

Ejercicio con funciones

17

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\\", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

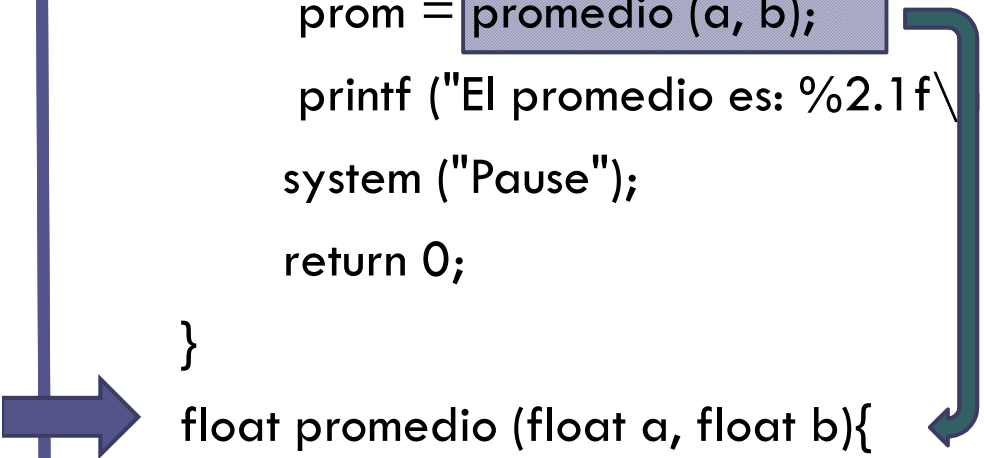


Variable	Valor
a	5
b	10
prom	

Ejercicio con funciones

18

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\\", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```



Variable	Valor
a	5
b	10
prom	
a	5
b	10

Ejercicio con funciones

19

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	
a	5
b	10
prom	

Ejercicio con funciones

20

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	
a	8
b	10
prom	

Ejercicio con funciones

21

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	
a	8
b	13
prom	



Ejercicio con funciones

22

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	
a	8
b	13
prom	10.5

Ejercicio con funciones

23

```
float promedio (float a, float b);
```

```
int main(){
```

```
    float a = 5, b=10 , prom;
```

```
    prom = promedio (a, b);
```

10.5

```
    printf ("El promedio es: %2.1f\n", prom);
```

```
    system ("Pause");
```

```
    return 0;
```

```
}
```

```
float promedio (float a, float b){
```

```
    float prom;
```

```
    a = a + 3;
```

```
    b = b + 3;
```

```
    prom = (a + b)/2;
```


```
    return prom;
```

```
}
```

Variable	Valor
a	5
b	10
prom	
a	8
b	13
prom	10.5

Ejercicio con funciones

24



```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

} **10.5**

Variable	Valor
a	5
b	10
prom	10.5
a	8
b	13
prom	10.5

Ejercicio con funciones

25

```
float promedio (float a, float b);  
int main(){  
    float a = 5, b=10 , prom;  
    prom = promedio (a, b);  
    printf ("El promedio es: %2.1f\n", prom);  
    system ("Pause");  
    return 0;  
}  
float promedio (float a, float b){  
    float prom;  
    a = a + 3;  
    b = b + 3;  
    prom = (a + b)/2;  
    return prom;  
}
```

Variable	Valor
a	5
b	10
prom	10.5
a	8
b	13
prom	10.5

El promedio es 10.5
Presione cualquier tecla para continuar...

Variables locales y globales

26

□ **Variables Locales:**

- ▣ Se declaran dentro de la función y sólo están disponibles durante su ejecución.
- ▣ Se crean cuando se entra en ejecución una función y se destruyen cuando se termina.

□ **Variables globales:**

- ▣ Se declaran fuera de las funciones. Pueden ser utilizadas por todas las funciones.
- ▣ Existen durante toda la vida del programa.

Ejercicio

27

- Escribir una función que se llame maximo que reciba dos número por parámetros y que regrese el mayor de ellos.

- Escribir una función que reciba caracteres del teclado hasta recibir un espacio o un salto de línea (enter) y a continuación mostrar todos los caracteres en orden inverso.
 - ▣ Ejemplo:
 - Entrada: Hola
 - Salida: aloH

Ejercicio

28

- Escribir una función que tome como parámetros las longitudes de los tres lados de un triángulo (a , b , c) y devuelva el área del triángulo.

$$A = \sqrt{p(p-a)(p-b)(p-c)} \quad \text{Donde } p = \frac{a+b+c}{2}$$



FUNCIONES RECURSIVAS

Funciones recursivas

30



- Se llaman funciones recursivas a aquellas que se llaman a su mismas de forma repetida hasta que se cumpla alguna condición.
- Cada llamada implica el almacenamiento de variables de estado y otros parámetros

Ejemplo: Calcular la potencia de forma recursiva

31

```
int main(void){  
    int x,y, max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}  
  
int potencia (a, b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

x	\0
y	\0
max	\0

Ejemplo: Calcular la potencia de forma recursiva

32

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (a, b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	\0
max	\0

Ejemplo: Calcular la potencia de forma recursiva

33

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

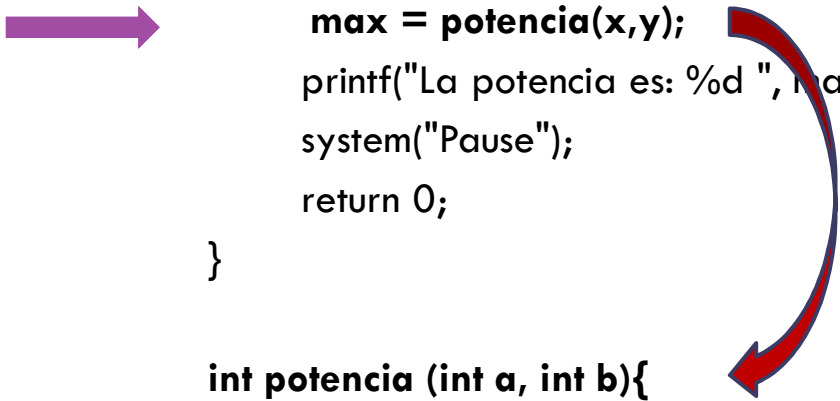
int potencia (a, b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	3
max	\0

Ejemplo: Calcular la potencia de forma recursiva

34

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}  
  
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```



x	2
y	3
max	\0

Ejemplo: Calcular la potencia de forma recursiva

35

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

➔

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

x	2
y	3
max	\0
a	2
b	3

1 {

Ejemplo: Calcular la potencia de forma recursiva

36

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

➡

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

x	2
y	3
max	\0
a	2
b	3

1 {

Ejemplo: Calcular la potencia de forma recursiva

37

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

1 {

x	2
y	3
max	\0
a	2
b	3

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

38

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

➔

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	3
max	\0
a	2
b	3
a	2
b	2

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

39

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

➡

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

x	2
y	3
max	\0
a	2
b	3
a	2
b	2

Ejemplo: Calcular la potencia de forma recursiva

40

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```



x	2
y	3
max	\0
a	2
b	3
a	2
b	2

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

41

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

➔ **int potencia (int a, int b){**
 if (b < 1)
 return 1;
 return a * potencia (a, b-1);
}

	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1

Ejemplo: Calcular la potencia de forma recursiva

42

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

→

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1

Ejemplo: Calcular la potencia de forma recursiva

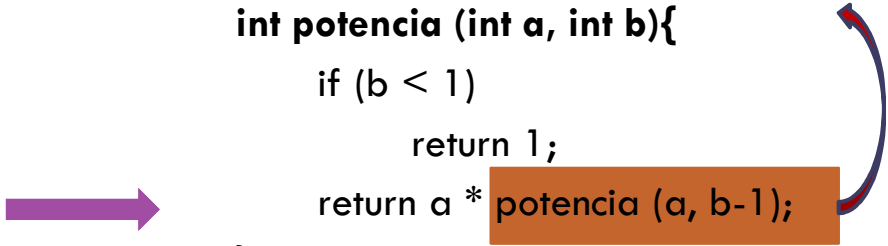
43

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```



	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

44

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

➔ **int potencia (int a, int b){**
 if (b < 1)
 return 1;
 return a * potencia (a, b-1);
}

	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1
4	a	2
	b	0

Ejemplo: Calcular la potencia de forma recursiva

45

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

→

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

1	a	2
	b	3
	a	2
	b	2
2	a	2
	b	1
3	a	2
	b	0
4	a	2
	b	0

Ejemplo: Calcular la potencia de forma recursiva

46

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1
4	a	2
	b	0

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

47

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	3
max	\0
a	2
b	3
a	2
b	2
a	2
b	1
a	2
b	0

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

48

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

→

2 1

	x	2
	y	3
	max	\0
1	a	2
	b	3
2	a	2
	b	2
3	a	2
	b	1

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

49

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	3
max	\0
a	2
b	3
a	2
b	2
a	2
b	1

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

50

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```



Diagram illustrating the recursive call: 2×2 . The first '2' is under 'a' and the second '2' is under 'potencia(a, b-1)'. A bracket connects them to the multiplication result.

x	2
y	3
max	\0
a	2
b	3
a	2
b	2

Diagram illustrating the recursive call stack. The first call (a=2, b=3) is grouped with a bracket labeled '1'. The second call (a=2, b=2) is grouped with a bracket labeled '2'.

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

51

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}
```

```
int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```

x	2
y	3
max	\0
a	2
b	3
a	2
b	2

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

52

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

→

2 4

x	2
y	3
max	\0
a	2
b	3

08/11/2013

Ejemplo: Calcular la potencia de forma recursiva

53

```
int main(void){  
    int x,y int max;  
  
    x = 2;  
    y = 3;  
  
    max = potencia(x,y);  
    printf("La potencia es: %d ", max);  
    system("Pause");  
    return 0;  
}
```

```
int potencia (int a, int b){  
    if (b < 1)  
        return 1;  
    return a * potencia (a, b-1);  
}
```

2

4

1 {

x	2
y	3
max	\0
a	2
b	3

08/11/2013

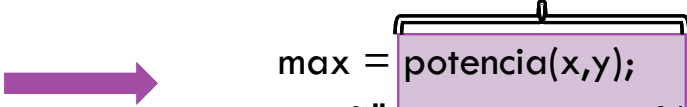
Ejemplo: Calcular la potencia de forma recursiva

54

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;
    8
    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```



x	2
y	3
max	\0

Ejemplo: Calcular la potencia de forma recursiva

55

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```



x	2
y	3
max	8

Ejemplo: Calcular la potencia de forma recursiva

56

```
int main(void){
    int x,y int max;

    x = 2;
    y = 3;

    max = potencia(x,y);
    printf("La potencia es: %d ", max);
    system("Pause");
    return 0;
}

int potencia (int a, int b){
    if (b < 1)
        return 1;
    return a * potencia (a, b-1);
}
```



x	2
y	3
max	8

La potencia es:8 Presione cualquier tecla para continuar...

Ejercicios

57

- Haz un programa con funciones recursivas que calcule el factorial de un número n ingresado desde teclado.

▣ Ej. $N = 5$

5!	= 4! * 5
4!	= 3! * 4
3!	= 2! * 3
2!	= 1! * 2
1!	= 0! * 1
0!	= 1

