Funciones
2^{da} parte

Nievas Martin

11 de agosto de 2020

```c
#include <stdio.h>

int func(int a, int b)
{
  int res;

  if (a > b)
    res = a;
  else
    res = b;

  return res;
}


int main(void)
{
  int num1 = 10;
  int num2 = 30;

  /** Llamado a la función */
  printf("%d\n",func(num1,num2));

  return 0;
}
```

```c
#include <stdio.h>

int func(int a, int b)
{
  int res;

  if (a > b)
    res = a;
  else
    res = b;

  return res;
}


int main(void)
{
  int num1 = 10;
  int num2 = 30;

  /** Llamado a la función */
  printf("%d\n",func(num1,num2));

  return 0;
}
```

```
30
```

```c
#include <stdio.h>

int max(int a, int b)
{
  int res;

  if (a > b)
    res = a;
  else
    res = b;

  return res;
}


int main(void)
{
  int num1 = 10;
  int num2 = 30;

  /** Llamado a la función */
  printf("%d\n", max(num1,num2));

  return 0;
}
```

```c
#include <stdio.h>

int max(int a, int b)
{
  int res;

  if (a > b)
    res = a;
  else
    res = b;

  return res;
}


int main(void)
{
  int num1 = 10;
  int num2 = 30;

  /** Llamado a la función */
  printf("%d\n", max(num1,num2));

  return 0;
}
```

```
30
```

Prototipos

Prototipos

```c
#include <stdio.h>

int max(int x, int y);

int main(void)
{









  return 0;
}
```

.

Prototipos

```c
#include <stdio.h>

int max(int x, int y);

int main(void)
{









  return 0;
}

int max(int x, int y){

  int max_val;

  if (x > y)
  {
    max_val = x;
  }
  else
  {
    max_val = y;
  }

  return max_val;
}
```

Prototipos

```c
#include <stdio.h>

int max(int x, int y);

int main(void)
{
  int num1;
  int num2;
  int mayor;

  printf("Ingrese dos números: ");
  scanf("%d %d", &num1, &num2);
  mayor = max(num1, num2);
  printf("El mayor es: %d\n", mayor);

  return 0;
}

int max(int x, int y){

  int max_val;

  if (x > y)
  {
    max_val = x;
  }
  else
  {
    max_val = y;
  }

  return max_val;
}
```

Prototipos

```c
#include <stdio.h>

int max(int, int);

int main(void)
{
  int num1;
  int num2;
  int mayor;

  printf("Ingrese dos números: ");
  scanf("%d %d", &num1, &num2);
  mayor = max(num1, num2);
  printf("El mayor es: %d\n", mayor);

  return 0;
}

int max(int x, int y){

  int max_val;

  if (x > y)
  {
    max_val = x;
  }
  else
  {
    max_val = y;
  }

  return max_val;
}
```

Prototipos

```c
#include <stdio.h>

int max(int, int);

int main(void)
{
  int num1;
  int num2;
  int mayor;

  printf("Ingrese dos números: ");
  scanf("%d %d", &num1, &num2);
  printf("El mayor es: %d\n", max(num1, num2);

  return 0;
}

int max(int x, int y){

  int max_val;

  if (x > y)
  {
    max_val = x;
  }
  else
  {
    max_val = y;
  }

  return max_val;
}
```

Prototipos

```c
#include <stdio.h>

int max(int, int);

int main(void)
{
  int num1;
  int num2;
  int mayor;

  printf("Ingrese dos números: ");
  scanf("%d %d", &num1, &num2);
  printf("El mayor es: %d\n", max(num1, num2);

  return 0;
}

int max(int x, int y){

  int max_val;

  if (x > y)
    max_val = x;
  else
    max_val = y;

  return max_val;
}
```

Variables dentro de funciones

Variables

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());

  return 0;
}
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());

  return 0;
}
```

```
valor: 1
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());

  return 0;
}
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());

  return 0;
}
```

```
valor: 1
valor: 1
valor: 1
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  static int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());

  return 0;
}
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  static int a = 0;

  return ++a;
}

int main(void)
{

  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());
  printf("valor: %d\n", incrementar());

  return 0;
}
```

```
valor: 1
valor: 2
valor: 3
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  static int a = 0;

  return ++a;
}

int main(void)
{

  for(int i = 0; i < 10; i++)
    printf("valor: %d\n", incrementar());

  return 0;
}
```

Variables

```c
#include <stdio.h>

int incrementar(void)
{
  static int a = 0;

  return ++a;
}

int main(void)
{

  for(int i = 0; i < 10; i++)
    printf("valor: %d\n", incrementar());

  return 0;
}
```

```
valor: 1
valor: 2
valor: 3
valor: 4
valor: 5
valor: 6
valor: 7
valor: 8
valor: 9
valor: 10
```

Factorial

Factorial

$$n! = 1 * \cdots * (n-2) * (n-1) * n$$

Factorial

$$n! = 1 * \cdots * (n-2) * (n-1) * n$$

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

Factorial

```c
#include <stdio.h>

int main(void)
{
  int factorial = 1;
  int cont;
  int num = 3;

  for (cont = num; cont >= 1; --cont)
    factorial *= cont;

  printf("factorial %d!: %d\n", num, factorial);

  return 0;
}
```

Factorial

```c
#include <stdio.h>

int main(void)
{
  int factorial = 1;
  int cont;
  int num = 3;

  for (cont = num; cont >= 1; --cont)
    factorial *= cont;

  printf("factorial %d!: %d\n", num, factorial);

  return 0;
}
```

```
factorial 3!: 6
```

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

$$n! = n * (n-1)!$$

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

$$n! = n * (n-1)!$$

Ejemplo:

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

$$n! = n * (n-1)!$$

Ejemplo:

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

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

$$n! = n * (n-1)!$$

Ejemplo:

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

$$5! = 5 * (4 * 3 * 2 * 1)$$

Factorial

$$n! = n * (n-1) * (n-2) * \cdots * 1$$

$$n! = n * (n-1)!$$

Ejemplo:

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

$$5! = 5 * (4 * 3 * 2 * 1)$$

$$5! = 5 * (4!)$$

Factorial - Recursivo

```c
#include <stdio.h>

int factorial(int);

int main(void)
{
  int num;

  for (num = 0; num < 10; num++)
    printf("%d! = %d\n", num, factorial(num));

  return 0;
}

int factorial(int number)
{
  if (number <= 1)
  {
    return 1;
  }
  else
  {
    return (number * factorial(number - 1));
  }
}
```

Factorial - Recursivo

```
0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
```

Factorial - Recursivo

```c
#include <stdio.h>

int factorial(int);

int main(void)
{
  int num;

  for (num = 0; num < 20; num++)
    printf("%d! = %d\n", num, factorial(num));

  return 0;
}

int factorial(int number)
{
  if (number <= 1)
  {
    return 1;
  }
  else
  {
    return (number * factorial(number - 1));
  }
}
```

Factorial - Recursivo

```
0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
11! = 39916800
12! = 479001600
13! = 1932053504
14! = 1278945280
15! = 2004310016
16! = 2004189184
17! = -288522240
18! = -898433024
19! = 109641728
```

Factorial - Recursivo

```c
#include <stdio.h>

unsigned long long int factorial(unsigned int);

int main(void)
{
  unsigned int num;

  for (num = 0; num < 20; num++)
    printf("%u! = %llu\n", num, factorial(num));

  return 0;
}

unsigned long long int factorial(unsigned int number)
{
  if (number <= 1)
  {
    return 1;
  }
  else
  {
    return (number * factorial(number - 1));
  }
}
```

Factorial - Recursivo

```
0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
11! = 39916800
12! = 479001600
13! = 6227020800
14! = 87178291200
15! = 1307674368000
16! = 20922789888000
17! = 355687428096000
18! = 6402373705728000
19! = 121645100408832000
```

Fibonacci

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

$$\mathit{fibonacci}(0) = 0$$

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

$$\mathit{fibonacci}(0) = 0$$

$$\mathit{fibonacci}(1) = 1$$

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

$$\mathit{fibonacci}(0) = 0$$

$$\mathit{fibonacci}(1) = 1$$

$$\mathit{fibonacci}(n) = \mathit{fibonacci}(n-1) + \mathit{fibonacci}(n-2)$$

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

$$fibonacci(0) = 0$$

$$fibonacci(1) = 1$$

$$fibonacci(n) = fibonacci(n-1) + fibonacci(n-2)$$

Implementación:

Fibonacci

$$0, 1, 1, 2, 3, 5, 8, 13, 21,$$

$$fibonacci(0) = 0$$

$$fibonacci(1) = 1$$

$$fibonacci(n) = fibonacci(n-1) + fibonacci(n-2)$$

Implementación: A cargo del alumno

mnievas@frc.utn.edu.ar

Consultas
Martes
Edificio Salcedo 10:00 - 13:00