CS1112: Programación II

Unidad 2: Funciones y recursividad

Sesión de Laboratorio - 3B

Profesores:
María Hilda Bermejo **mbermejo@utec.edu.pe**
Estanislao Contreras **econtreras@utec.edu.pe**
Jorge Villavicencio **jvillavicencio@utec.edu.pe**
Edson Mendiolaza **emendiolaza@utec.edu.pe**
Ian Paul Brossard **ibrossard@utec.edu.pe**
Jose Chavez **jchaveza@utec.edu.pe**
Julio Yarasca **jyarascam@utec.edu.pe**
Percy Quevedo **pquevedo@utec.edu.pe**
Wilder Nina **wnina@utec.edu.pe**
José Fiestas **jfiestas@utec.edu.pe**

Material elaborado por:
Maria Hilda Bermejo, Jaime Farfán

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Índice:

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

2.2 Recursividad

Logro de la sesión:

Al finalizar la sesión, los alumnos:

- Entienden el concepto de variables globales.
- Comprenden el concepto de transferencia por valor y por referencia.
- Usan funciones recursivas y no-recursivas

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Recursividad

Las funciones recursivas se componen de:

–Caso base: una solución simple para un caso particular (puede haber más de un caso base).

–Caso recursivo: una solución que involucra volver a utilizar la función original, con parámetros que se acercan más al caso base.

Ejemplo 1:

El factorial de un número.

$$Fact(n) = \begin{cases} 1 & si\ n = 0; \\ n.Fact(n-1) & si\ n > 0 \end{cases}$$

Caso base

Caso recursivo

Ejemplo 1:

Veamos cómo funciona la recursividad, hallando el factorial de 5

```
fact(5) = 5 * fact(4)
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
                          3 * fact(2)
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
                          3 * fact(2)
                                2 * fact(1)
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
                          3 * fact(2)
                                2 * fact(1)
                                      1 * fact(0)
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
                          3 * fact(2)
                                2 * fact(1)
                                      1 * 1
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                    4 * fact(3)
                          3 * fact(2)
                                2 * fact(1)
                                      1
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                  4 * fact(3)
                        3 * fact(2)
                              2 * fact(1)
                                    1
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                  4 * fact(3)
                        3 * fact(2)
                              2 * 1
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)
                  4 * fact(3)
                        3 * 2
```

Ejemplo 1:

```
fact(5) = 5 * fact(4)

                  4 * 6
```

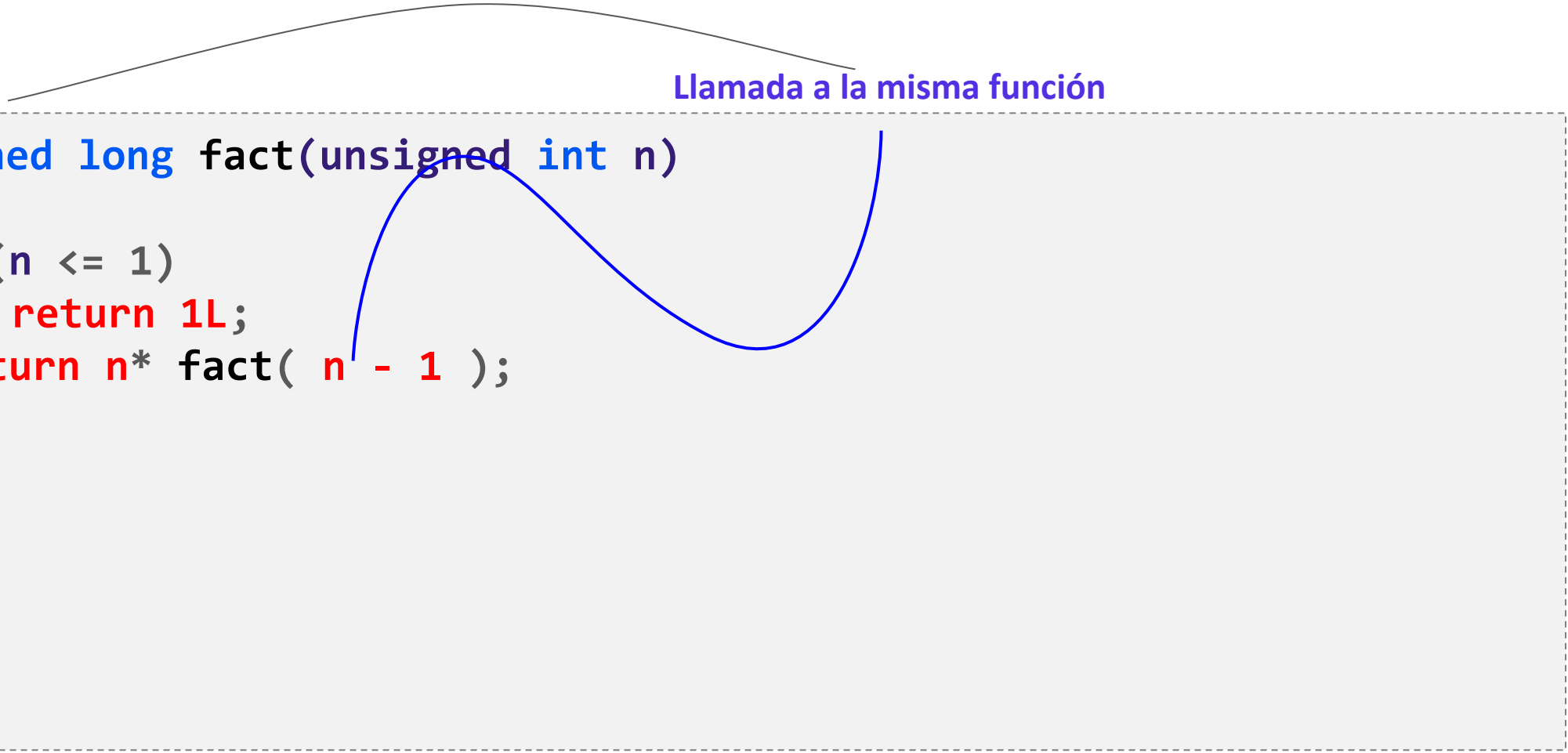
Ejemplo 1:

```
fact(5) = 5 * 24
```

Ejemplo 1:

```
fact(5) = 120
```

Recursión:




```
unsigned long fact(unsigned int n)
{
    if(n <= 1)
        return 1L;
    return n* fact( n - 1 );
}
```

Factorial

Factorial - Iterativo

```cpp
#include <iostream>
using namespace std;

unsigned  long factorial(unsigned int  n)
{unsigned long f;

 f=1;
 for(int i=2;i<=n; i++)
   f*=i;
 return f;
}

int main()
{
 unsigned int n;
 cout << "Numero: ";
 cin >> n;
 cout << "Factorial ("<< n <<") = ";
 cout << factorial(n);
 return 0;
}
```

Factorial - Recursivo

```cpp
#include <iostream>
using namespace std;

unsigned long factorial(unsigned int  n)
{
 if (n<=1)
   return 1L;
 return (n * factorial(n-1));
}

int main()
{
 unsigned int n;
 cout << "Numero: ";
 cin >> n;
 cout << "Factorial ("<< n <<") = ";
 cout << factorial(n);
 return 0;
}
```

Ejemplo 2:

Hallar recursivamente el producto de dos números enteros positivos.

$$A.B = \begin{cases} A & \text{si } B = 1; \\ A + A.(B-1) & si\ B > 1; \end{cases}$$

Ejemplo 3:

Hallar recursivamente la potencia entera n (n >= 0) de un número real X no nulo

$$X^n = \begin{cases} 1 & si\ n = 0; \\ X.X^{(n-1)} & si\ n > 0 \end{cases}$$

Ejercicios

Ejercicio 2:

Genere una función recursiva que calcule la suma de cuadrados de los números de 1 a n, donde n es ingresado por el usuario.
Genere una secuencia aleatoria de 10 enteros entre 1 y 30. Utilice la función anterior para generar una lista de valores acumulados de los cuadrados de la lista original (suma de prefijos).

```
lista original: 5 1 18 2 4 6 21 9 10 15 1
suma de prefijos: 25 26 350 354 370 406 847 928 1028 1253 1254
```

Ejercicio 3:

Hallar recursivamente el máximo elemento de un vector.

Ejercicio 1:

Hallar el monto final de un monto inicial (capital) despues de t años, a un interés de i % anual, de acuerdo a la siguiente fórmula:
monto_final=monto_inicial*(1+i/100)
, donde i es el interés anual
Implemente dos funciones para el cálculo, una para el método recursivo y otra para el no recursivo.

```
Monto inicial:
1000
Interés anual:
10
monto final(no recursivo): 2593.74
monto final(recursivo): 2593.74
```

Lo aprendido hoy:

- La recursividad es una característica que permite a un subprograma invocarse a sí mismo.
- Cualquier proceso iterativo puede expresarse en forma recursiva y viceversa.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Bibliografía:

Deitel. P.J. and Deitel. H. M. (2016) C++ How to Program, Prentice Hall.

Stroustrup, Bjarne (2013). The C++ Programming Language, 4th Addison-Wesley.

Eckel, Bruce, 2000. Thinking in C++, Vol 1: Introduction to Standard C++, 2nd Edition, Prentice Hall

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

¡Nos vemos en la siguiente clase!

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA