Semana-06 – Blender

FUNCIONES

Temario

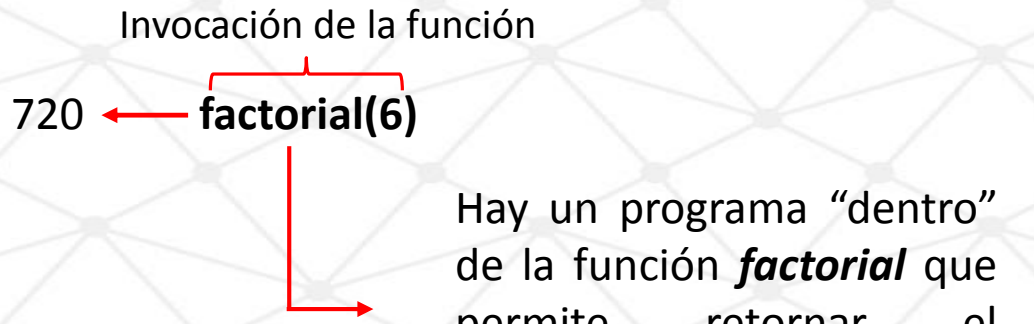
- Implementación de funciones con retorno y sin retorno en C++

Función en la programación - CONCEPTO

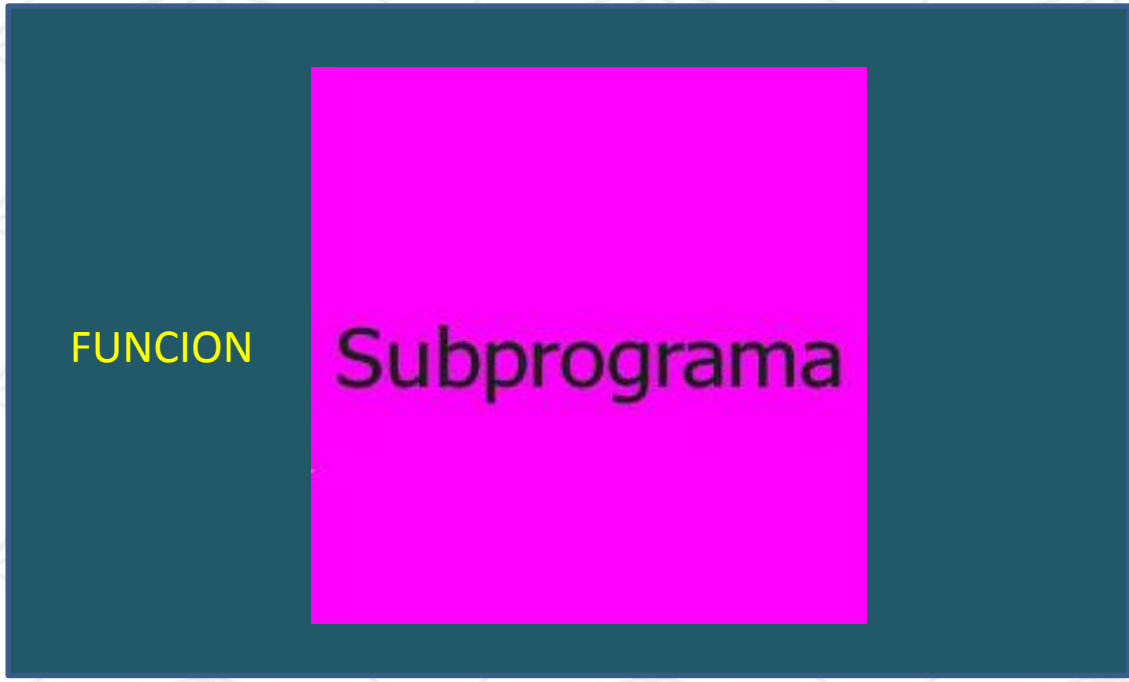
Concepto:

Es una subrutina o subprograma encapsulado, el cual permite resolver una tarea específica y que puede ser invocada (llamada a ejecución) desde otras partes de un programa tantas veces como se desee.

Ejemplo:



Hay un programa "dentro" de la función ***factorial*** que permite retornar el factorial de 6 (de cualquier número en realidad)



Construcción de funciones en C++

Partamos de un Ejemplo:

Implementar una función que retorne el valor máximo de dos números reales (No usar la función max)

¿Qué se espera de la función cuando sea invocada?

Se espera que la función cuando sea invocada y reciba dos números reales, por ejemplo **13.5** y **24,** retorne el mayor de ellos, osea **24**

Por lo tanto debería haber un programa dentro de la función que permita lograr tal objetivo para cualquier par de números enviados.

Construcción de funciones en C++

Formato para construir una función con retorno de valor :

Puede ser: **int**, **double**, **float**, **char**, ….

Nombre de la función

```
tipo de retorno     nombre(argumento 1, argumento 2, …… argumento n)
{
                        .
                        .
                        .
                        .
     instrucción que permita retornar el resultado
}
```

declaración o **definición** de la función

Los argumentos de la función recibirán los valores enviados en el momento en que la función sea invocada, estos valores serán procesados y al final la función retornará cuando mucho **un solo valor.**

Construcción de funciones en C++

¿Donde se construyen las funciones?

```
#include <iostream>

using namespace std;

int main()
{


   return 0;
}
```

Las funciones se construyen en esta zona (fuera de la función **main())**

Solución del ejemplo:

```
#include <iostream>
using namespace std;
float  encontrarMax(float x, float y)
{
    float numMax;
    if(x>=y)
        numMax = x;
    else
        numMax = y;

    return numMax; //instrucción de
                   //retorno
}
int main()
{

   return 0;
}
```

Función implementada con 2 argumentos de entrada

```
float  encontrarMax(float x, float y)
```

Indica que la función retornará un valor float

Para que el programa de la función se ejecute esta debe ser *llamada* o *invocada* en la **función main**

Invocación de la función:

```
int main()
{
  float num1,num2,mayor;

  cout<<"Ingrese 2 numeros: ";
  cin>>num1>>num2;

  mayor = encontrarMax(num1, num2);

  cout<<"El numero mayor es: "<<mayor;

  return 0;
}
```

Instrucción de invocación de la función

Resultado de la ejecución

```
Ingrese 2 numeros: 13.5 24
El numero mayor es: 24
--------------------------------
Process exited after 2.691 seconds with return value 0
Press any key to continue . . .
```

1 Cuando se invoca a la función, los valores almacenados en las vbles. *num1* y *num2* son transmitidos a los argumentos de entrada (*x* e *y*) presentes en la definición de la función

num1 13.5 num2 24

```
float  encontrarMax(float x, float y)
{
        float numMax;
        if(x>=y)
           numMax = x;
        else
           numMax = y;

        return numMax;
}
```

2 La función ejecuta sus instrucciones con los valores enviados. La palabra ***return*** indica que la función termina su aplicación retornando un valor. En este caso retornará el valor 24 ya que es el mayor.

3 El valor retornado por la función es enviado a la vble. ***mayor*** creada en la función main

TIPS importantes

```
float  encontrarMax(float x, float y)
{
       float numMax;

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

       return numMax;
}
```

Variables locales

Toda variable creada en la declaración y/o cuerpo de una función es llamada ***variable local***.
Una variable local **SOLAMENTE** existe dentro del contexto la función.

```
int main()
{
  float num1,num2,mayor;

  cout<<"Ingrese 2 numeros: ";
  cin>>num1>>num2;

  mayor = encontrarMax(num1, num2);

  cout<<"El valor de x es: "<<numMax;

  return 0;
}
```

Error: la vble. *numMax* **NO** existe en la función main()

Problema:

Escriba un programa que permita leer uno por uno el valor 4 temperaturas en grados Fahrenheit, y muestre después de cada lectura el valor correspondiente en grados Celsius.

Ejemplo:

```
Introduzca una temperatura en grados fahrenheit: 13
Su equivalente en grados Celsius es: -10.5556
Introduzca una temperatura en grados fahrenheit: 345.5
Su equivalente en grados Celsius es: 174.167
Introduzca una temperatura en grados fahrenheit: 132
Su equivalente en grados Celsius es: 55.5556
Introduzca una temperatura en grados fahrenheit: 10
Su equivalente en grados Celsius es: -12.2222
```

Solución

```
#include <iostream>

using namespace std;

float  convertirTemperatura(float temp)
{
   return (5.0/9)*(temp - 32.0);  //retorna el resultado de la
                                  //conversión (fórmula)
}

int main()
{
    float far;

    for(int i=1;i<=4;i++) //repite bloque 4 veces
    {
        cout<<"Introduzca una temperatura en grados fahrenheit: ";
        cin>>far;

        cout<<"\nSu equivalente en grados Celsius es: ";
        cout<<convertirTemperatura(far)<<endl<<endl;
    }
    return 0;
}
```

La función es invocada 4 veces

TIP importante

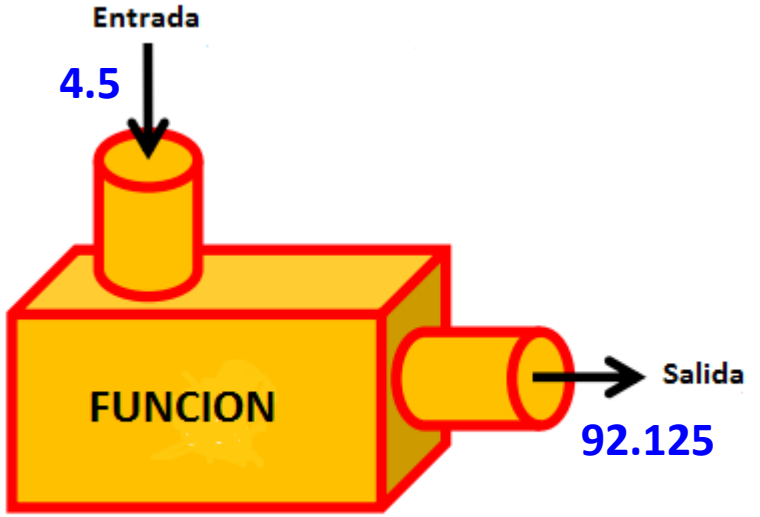
Una función con retorno puede verse de manera **figurativa** como una “máquina”:

Ejemplo:

```
float  funcion(float x)
{
   return  pow(x,3) + 1;
}

int main()
{
   cout<<funcion(4.5);

   return 0;
}
```

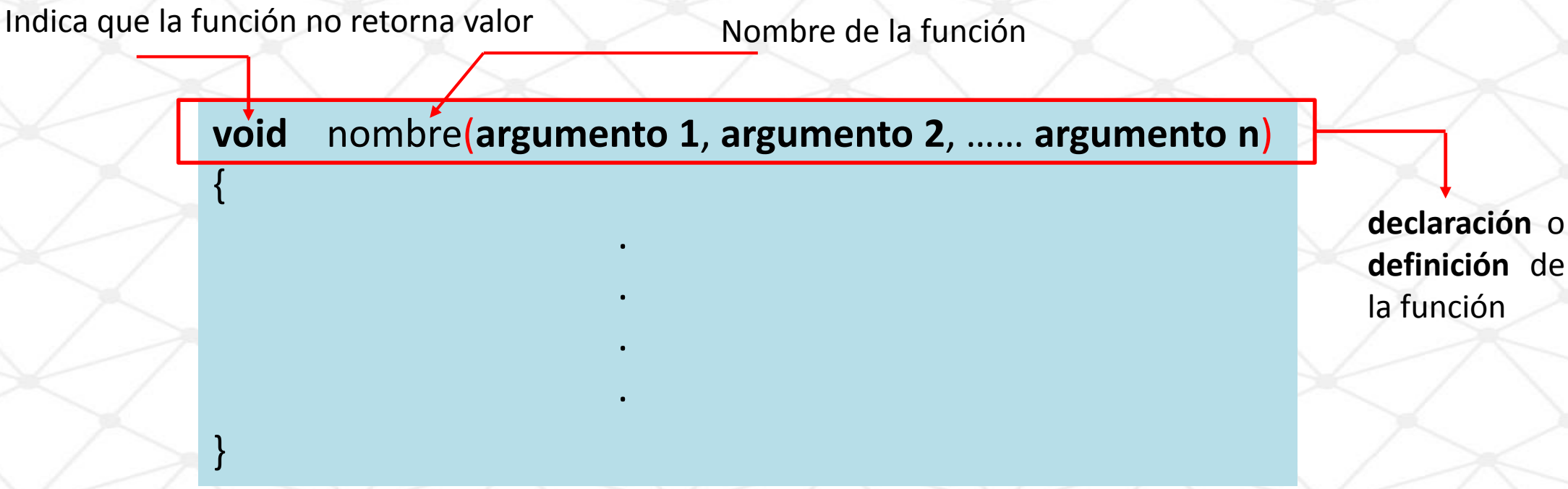


Resultado de la ejecución

```
92.125
--------------------------------
Process exited after 0.1436 seconds with return value 0
Press any key to continue . . .
```

Construcción de funciones en C++

Formato para construir una función sin retorno de valor :



Los argumentos de la función recibirán los valores enviados en el momento en que la función sea invocada, estos valores serán procesados, se mostrará un resultado mas NO será retornado.

Construcción de funciones en C++

Ejemplo:

```
#include <iostream>
using namespace std;
void  encontrarMax(float x, float y)
{
    float numMax;
    if(x>=y)
        numMax = x;
    else
        numMax = y;

    cout<<"El numero mayor es:"<<numMax;

}
int main()
{

   return 0;
}
```

Función implementada con 2 argumentos de entrada

```
void  encontrarMax(float x, float y)
```

Indica que la función No retornará un valor

En este caso el resultado es mostrado en pantalla por la función, mas no es retornado por la misma

Invocación de la función:

```
int main()
{
  float num1,num2;

  cout<<"Ingrese 2 numeros: ";
  cin>>num1>>num2;

  encontrarMax(num1, num2);

  //ya no es necessária una instrucción
  //para mostrar el mayor.

  return 0;
}
```

Instrucción de invocación de la función

Resultado de la ejecución

```
Ingrese 2 numeros: 13.5 24
El numero mayor es: 24
--------------------------------
Process exited after 2.691 seconds with return value 0
Press any key to continue . . .
```

3 Como la función no retorna ya no es necesaria una variable adicional en main()

1 Cuando se invoca a la función, los valores almacenados en las vbles. *num1* y *num2* son transmitidos a los argumentos de entrada (*x* e *y*) presentes en la definición de la función

num1 13.5 num2 24

```
void   encontrarMax(float x, float y)
{
      float numMax;
      if(x>=y)
            numMax = x;
      else
            numMax = y;

      cout<<"El numero mayor es:"<<numMax;
}
```

2 La función ejecuta sus instrucciones con los valores enviados. La función termina su aplicación mostrando el resultado (no lo retorna). En este caso mostrará el valor 24 ya que es el mayor.

TIPS importantes

Tener cuidado al invocar a una función sin retorno de valor:

```
void   encontrarMax(float x, float y)
{
     float numMax;
     if(x>=y)
          numMax = x;
     else
          numMax = y;

     cout<<"El numero mayor es:"<<numMax;
}
```

```
int main()
{
  float num1,num2, mayor;

  cout<<"Ingrese 2 numeros: ";
  cin>>num1>>num2;

  mayor = encontrarMax(num1, num2);

  return 0;
}
```

Error: Debido a que la función no retorna, no estaría correcto asignar el resultado a una variable

TIPS importantes

No es obligatorio que siempre haya argumentos de entrada en una función

Ejemplo:

Construir una función que imprima en pantalla un menú de opciones

```
#include <iostream>

using namespace std;

void imprimeMenu() {
   cout<<"MENU\n";
   cout<<"[1]Sumar\n";
   cout<<"[2]Restar\n";
   cout<<"[3]Multiplicar\n";
   cout<<"[4]Dividir\n";
}

int main() {

    imprimeMenu();//invocación

    return 0;
}
```

En este caso, la función no recibe argumentos de entrada y no retorna valor, solo imprime un texto en pantalla.

Resultado de la ejecución

```
MENU
[1]Sumar
[2]Restar
[3]Multiplicar
[4]Dividir

--------------------------------
Process exited after 0.1567 seconds with return value 0
Press any key to continue . . .
```