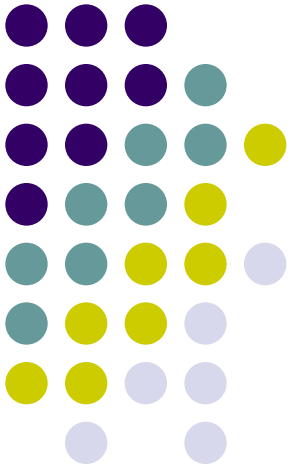
http://webs.um.es/frgarcia/teaching.htm

ALGORITMOS Y ESTRUCTURAS DE DATOS



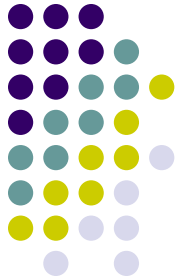
PRÁCTICA. TEMAS 2 Y 3

SEMINARIO DE C++

SESIÓN 1

Francisco García Sánchez
frgarcia@um.es
Despacho 2.31

INDICE DE CONTENIDOS

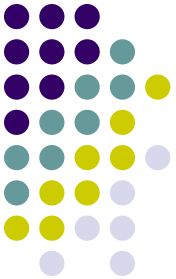


- Práctica: Tienda On-Line
- Introducción a C++
  - Características generales
  - Entrada/salida estándar
  - Variables y tipos de datos
  - Funciones
  - Memoria dinámica
  - Definición de clases
  - Implementación de los métodos
  - Constructores y destructores
  - Objetos en memoria dinámica
  - ‘`cin`’ en la práctica
- Ejercicios

Práctica de los temas 2 y 3

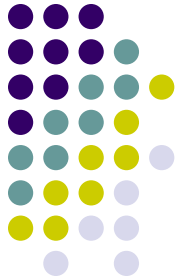
TIENDA ON-LINE

Tienda On-Line



- Las tiendas ofrecen un amplio catálogo de productos, cuanto más grande mejor
- Afinar la búsqueda de los productos es un aspecto esencial en estas aplicaciones:
  - mucho tiempo en responder, o
  - no ofrece al usuario lo que busca
  - ➲ Potenciales clientes huyen hacia las páginas de la competencia
- **OBJETIVO** ➔ Crear un sistema capaz de almacenar información de productos y realizar consultas eficientes sobre los mismos

Tienda On-Line



- Comandos admisibles:
  - **Insertar nuevo producto**: identificador + nombre + descripción + precio

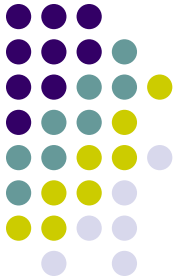
Entrada

```
insertar 84021349837
Pascuard Bell Viseo230 23" Full HD
Monitor Full HD de 23" con un precio increíble . Alta definición . Acabado elegante .
99.95
```

Salida

```
462 productos
```

Tienda On-Line



- Comandos admisibles:
  - **Buscar por palabras**: listar todos los productos que contienen las palabras que se le pasan como parámetro
    - En *nombre* o *descripción*
    - Independiente de mayúsculas/minúsculas

Entrada

```
palabras monitor bell alta definición
```

Salida

```
1. Pascuard Bell Viseo230 23" Full HD (84021349837)
Monitor Full HD de 23" con un precio increíble . Alta definición . Acabado elegante .
99.95 euros
Total: 1 productos
```

Tienda On-Line

- Comandos admisibles:
  - **Buscar por precios**: listar todos los productos cuyo precio se encuentre en el rango que se pasa como parámetro

Entrada

```
precios 80 99.99
```

Salida

```
1. Pascuard Bell Viseo230 23" Full HD (84021349837)
Monitor Full HD de 23" con un precio increíble . Alta definición . Acabado elegante .
99.95 euros
Total: 1 productos
```

Tienda On-Line

- Comando **opcional**:
  - **Eliminar**: elimina un producto dado su identificador

Entrada

```
eliminar 8402134983 7
```

Salida

```
461 productos
```

Tienda On-Line

- Comando **opcional**:
  - **Producto**: Dado un identificador de producto, realiza una búsqueda del mismo y muestra sus datos en la salida

Entrada

```
producto 8402134983 7
```

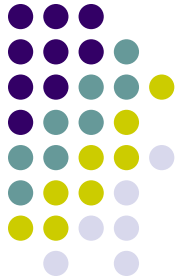
Salida

```
1. Pascuard Bell Viseo230 23" Full HD (8402134983 7)
Monitor Full HD de 23" con un precio increíble . Alta definición . Acabado elegante .
99.95 euros
Total: 1 productos
```

Tienda On-Line

- Documentación:
  - Portada
  - Análisis del problema
  - Diseño de la aplicación
  - Listado del código
  - Informe de desarrollo
  - Conclusiones y valoraciones personales

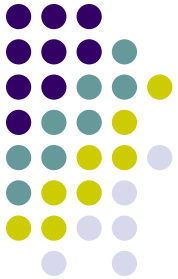
Tienda On-Line



- Evaluación:
  - **Obligatorio**:
    - Compilar sin errores + *Accepted* en Mooshak
    - Funcionar correctamente
    - Memoria de la práctica completa y sin faltas ortográficas
  - **Valoración**: análisis y diseño, modularidad, uso del lenguaje, seguimiento continuo, opcionales
  - Grupos de 2 alumnos
  - **Fecha de entrega**: 13 de diciembre de 2012

Tienda On-Line

- Calendario de sesiones
  - Sesión 1: 22 de octubre
  - Sesión 2: 29 de octubre
  - Sesión 3: 5 de noviembre
  - Sesión 4: 12 de noviembre
  - Sesión 5: 19 de noviembre
  - Sesión 6: 26 de noviembre
  - Sesión 7: 3 de diciembre
  - Sesión 8: 10 de diciembre

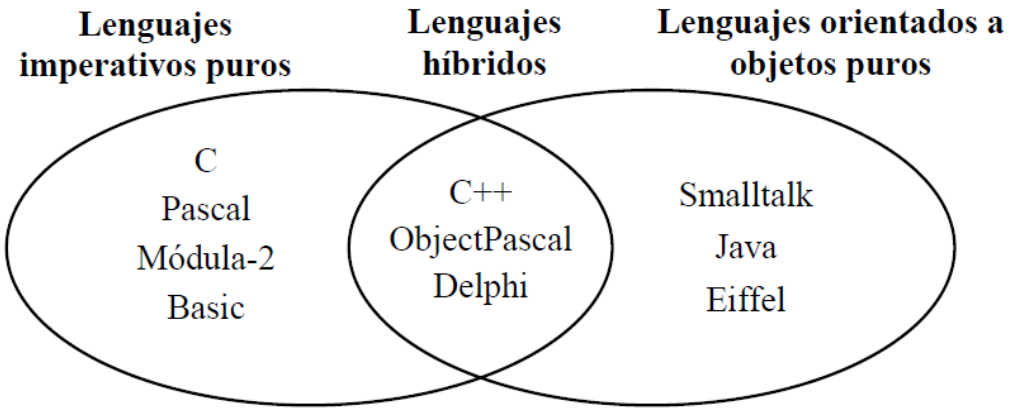


Sesion 1

INTRODUCCIÓN A C++

Introducción a C++

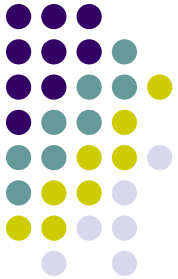
- Características generales:
  - Lenguaje híbrido



  - Programación orientada a objetos: CLASES
    - Clase ≈ Tipo abstracto de datos
    - Clase ≈ Módulo → encapsulación (funciones + datos) y ocultamiento (`private` & `public`)

Descargar: http://dis.um.es/~ginesgm/files/doc/aed/seminario2-1.pdf

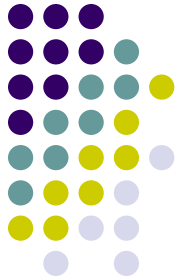
Introducción a C++



- Características generales:
  - C++ vs C:
    - Extensión ficheros de código: **.cpp**
    - Extensión ficheros de cabecera: **.hpp**
    - Compilación: **c++** o **g++**
    - Comentarios:

```
/* Esto es un comentario de C
   que se puede extender varias líneas */

// Y esto es un comentario de C++
// que se extiende hasta fin de línea
```

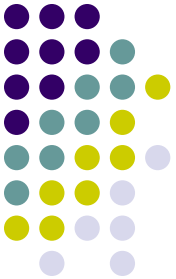
Introducción a C++



- Entrada/salida estándar:
  - Librería <**iostream**>
  - Variables:
    - **cin** → flujo de entrada estándar
    - **cout** → flujo de salida estándar
    - **cerr** → flujo de error estándar
  - Operadores:
    - **cin >> variable** → leer un valor de teclado y escribir en variable
    - **cout << expresion** → escribir la expresión en la salida estándar
    - **cerr << expresion** → igual, con la salida de error estándar

http://www.cplusplus.com/reference/iostream/

Introducción a C++

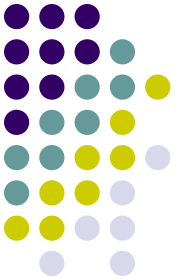


- Entrada/salida estándar:

```
#include <iostream>          // Ojo: notar que va sin .h
using namespace std;
int main ()
{
   int numero= 1;            // Esta inicialización es irrelevante
   char nombre[80];
   cout << "¿Como te llamas? ";
   cin >> nombre;
   cout << "Hola " << nombre << '\n';  // Se pueden poner varios <<
   cout << "Un número: ";
   cin >> numero;
   cout << "El C++ es " << numero << " veces mejor que el C.\n";
    /* Añadir por aquí */
}
```

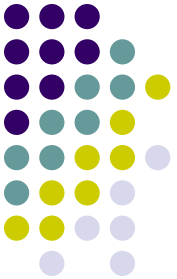
Compilación: `g++ -g -Wall nombreFichero.cpp -o a.out`

Introducción a C++



- Variables y tipos de datos:
  - Tipo de datos booleano: `bool`
    - Valores: `true` y `false`
    - Operadores: `and`, `or`, `not`, `xor`
  - Tipo de datos cadena de carácteres: `string`
    - `#include <string>`
    - `string variable = "valor variable";`
    - Métodos: length, size, substr, +=, compare, …
    - **http://www.cplusplus.com/reference/string/string/**
    - **http://anaturb.net/C/string_exapm.htm**

Introducción a C++



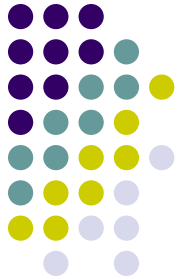
- Variables y tipos de datos:
  - Declaración de variables:
    - En cualquier punto del código
  - Declaración de constantes: `const`
    - Deben inicializarse siempre
    - Utilizar constantes `const` en lugar de `#define`
      - Constantes de tipo puntero

```
const char *nombre= "Pepito";
```

      - Parámetros constantes en funciones

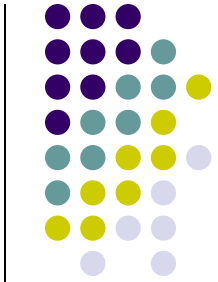
```
char * strcpy (char * destino, const char *origen)
```

Introducción a C++



- Variables y tipos de datos:
  - Espacios de nombres y visibilidad
    - Operador de resolución de visibilidad: '`::`'
      - Mismo nombre de variable que aparece en distintos ámbitos
    - Espacios de nombres: `namespace`
      - En lugar de declarar todos los datos y funciones como globales
      - `namespace ambito_a {…}` → `ambito_a::nombre`
      - Uso de espacio sin prefijo: **`using namespace`**
        - `cout`, `cin` y `cerr` definido en espacio de nombres `std` → `using namespace std;`

Introducción a C++



- Funciones:
  - Paso de parámetros por referencia: `tipo & nombre`

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

Llamada:

```
int i;
suma(2, 5, i);
```

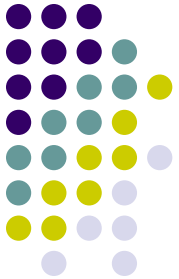
  - Variables de tipo referencia

```
int i= 1;
int &r= i;          // r referencia al mismo sitio que i
r++;
i++;
cout << r;
```

  - Parámetros por defecto

```
tipoDevuelto nombreFuncion (tipo1 nombre1, tipo2 nombre2, ...,
                            tipon-1 nombren-1= valorn-1, tipon nombren= valorn)
```

Introducción a C++



- Funciones:
  - **Sobrecarga de funciones**: dos o más funciones distintas pueden tener el mismo nombre
    - Los tipos de los parámetros deben ser distintos

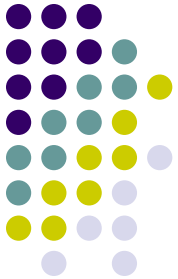
Ejemplo (pág. 7)

```
void saludar (void)
{
   cout << "Bienvenido, señor desconocido.";
}

void saludar (char *nombre)
{
   cout << "¡Hola " << nombre << "!";
}

void saludar (int n)
{
   cout << "Te saludo " << n << " veces.";
}
```

Introducción a C++

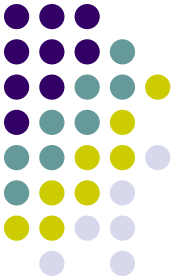
- Memoria dinámica:
  - **Operadores**: `new`, `new []`, `delete` y `delete []`
  - **`new` *`tipo`*** → crea nueva variable de tipo *`tipo`*. Devuelve un puntero a la variable creada.
  - **`new` *`tipo[tamaño]`*** → crea un array dinámico de tipo *`tipo`* y de *`tamaño`* elementos. Devuelve puntero al inicio del array.
  - **`delete puntero`** → borrar una variable apuntada por `puntero`.
  - **`delete [] puntero`** → borrar una array dinámico apuntado por `puntero`.
  - En general se recomienda usar **`new`** y **`delete`** en lugar de `malloc` y `free`.

Ejemplo (pág. 9): Reservar memoria para una matriz de tamaño **n** x **m**.



Sesion 2

INTRODUCCIÓN A C++

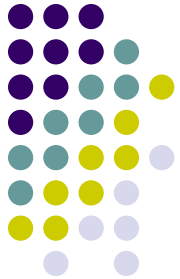


Introducción a C++

- Definición de clases:
  - ***Modelo imperativo***: programa ≡ sucesión de instrucciones secuenciales o iterativas
  - ***Modelo orientado a objetos***: programa ≡ conjunto de objetos que se comunican entre sí
  - **Objeto**: variable cuyo tipo es una clase

```
class persona                          // persona es una clase
{
   char nombre[80];
   long dni, telefono;
   int edad;

   void leer (FILE *f);
   void escribir (void);
};

persona p1;                            // p1 es un objeto de clase persona
p1.leer(f1);
p1.edad++;
p1.escribir();
```

Introducción a C++



- Definición de clases:
  - Nomenclatura:
    - ***Miembros de una clase***: atributos y operaciones de la clase (datos y funciones); p.ej. `dni` y `edad`
    - ***Métodos de una clase***: funciones definidas dentro de la clase; p.ej. `leer()` y `escribir()`
    - ***Mensaje***: invocación de un método sobre un objeto; p.ej. `p1.leer(f1)`
    - ***Objeto receptor***: objeto sobre el que se aplica un mensaje; p.ej. en `p1.leer(f1)` el receptor es `p1`.

Introducción a C++

- Definición de clases:
  - **Declaración:**

```
class nombre {
  ...
};
```

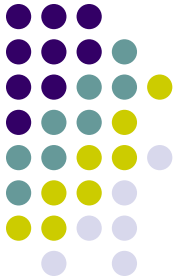
  - **Miembros públicos y privados:**

```
class nombre {
  private:
     ...        // Miembros privados
  public:
     ...        // Miembros públicos
};
```

  - **Declaración de los miembros:**

```
class conjuntoInt {
  private:
    int tamano;
    int datos[MAXIMO];
  public:
    void vaciar ();
    void insertar (int n);
    void suprimir (int n);
    bool miembro (int n);
};
```

Introducción a C++



- Implementación de los métodos:
  - Generalmente fuera de la definición de la clase
    - Operador de visibilidad: ‘: :’

```
void conjuntoInt::vaciar ()
{
    ...
}
```

```
void conjuntoInt::insertar (int n)
{
    ...
}
```

    - Definiendo el método ***todos*** los miembros de la clase son accesibles (como variables locales).
  - **Métodos in-line**:
    - Método dentro de la definición de la clase.
    - No aconsejable, a menos que sea trivial.

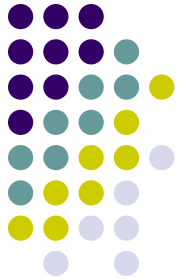
Introducción a C++

- Implementación de los métodos:
  - Clase conjunto de enteros con tamaño máximo limitado

Ejemplo página 5

Cambiar `#define` *por* `const int`

Introducción a C++



- Constructores y destructores:
  - **Constructor**: operación de inicialización de un objeto.
    - Por defecto, sin constructor, datos no se inicializan.
    - **Constructor** ≡ método con mismo nombre que clase
    - **Sobrecarga** permitida:

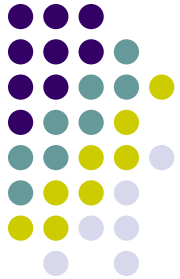
```
class conjuntoInt {
     ...
  public:
     conjuntoInt ()  {tamano= 0;} // Constructor de conjunto vacio
     conjuntoInt (int e1)         // Constructor de cjt. con 1 elem.
             {datos[0]= e1; tamano= 1;}
```

constructor por defecto

    - **Uso** constructores:

```
conjuntoInt cjt;     // Se inicializa con el constructor por defecto
conjuntoInt cjt(9);  // Constructor que incluye 1 elemento
```

Introducción a C++

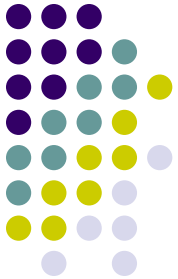


- Constructores y destructores:
  - **Destructor**: operación de eliminación de un objeto
    - Nombre del destructor: "**~NombreClase**"
    - No recibe parámetros; no devuelve ningún valor
    - Necesario si el objeto ha reservado memoria dinámica o abre algún fichero.
      - **Programador responsable de liberar/cerrar**

```
class conjuntoInt {
      ...
  public:
      ~conjuntoInt () {cout<<"Nada";} // Destructor, irrelevante aquí
     ...
};
```

```
cjt.~conjuntoInt();
```

Introducción a C++



- Objetos en memoria dinámica:
  - Igual a otros tipos de datos
    - Operadores: `new`, `new []`, `delete` y `delete []`
    - Crea objetos en memoria dinámica…necesario liberar memoria cuando objeto ya no usado.

```
nombreClase *obj1= new nombreClase;
```
→ Crear un nuevo objeto usando el constructor por defecto.

```
nombreClase *obj2= new nombreClase(p1, p2, ...);
```
→ Crear un nuevo objeto usando un constructor específico.

```
nombreClase *ad= new nombreClase[tamano];
```
→ Crear un array dinámico de objetos con el constructor por defecto.

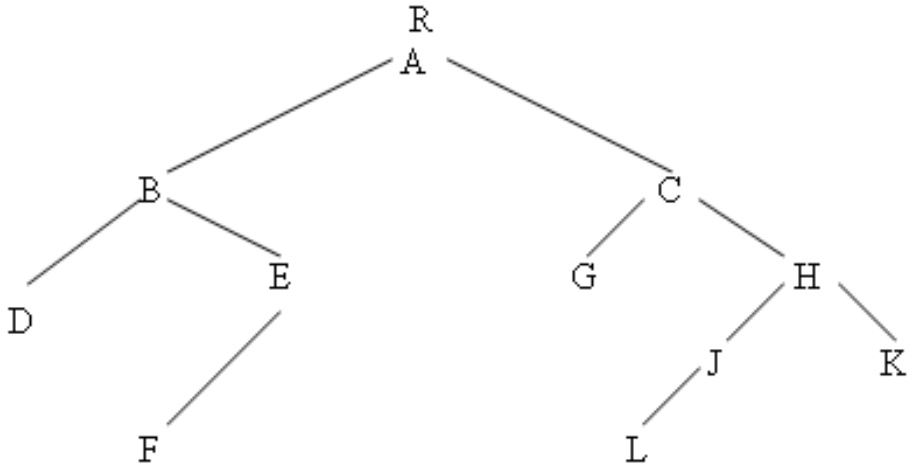
`delete obj1;` → Borra el objeto dinámico, usando el destructor.

`delete[] ad;` → Borra un array dinámico de objetos, usando el destructor.

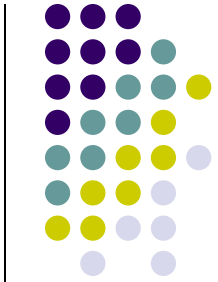
Introducción a C++

- Objetos en memoria dinámica:
  - Árbol binario de cadenas

Ejemplo página 8

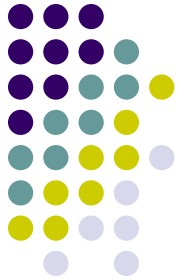


Introducción a C++



- **cin**:
  - Objeto de la clase '`istream`'
    - http://www.cplusplus.com/reference/iostream/istream/
    - `cin.eof()` – marca fin de fichero
    - `cin.get()` – obtener datos sin formato
  - Obtener cadena en una línea:
    - Librería <`string`>
    - `getline(istream& is, string& str)`

```
// getline with strings
#include <iostream>
#include <string>
using namespace std;

int main () {
  string str;
  cout << "Please enter full name: ";
  getline (cin,str);
  cout << "Thank you, " << str << ".\n";
}
```



Semana 1

EJERCICIOS

```
./a.out < entrada > salida
```

Ejercicios

- http://olimpiadas.inf.um.es/~mooshak/
  - Concurso: "**Preparación OIRM'13:programación**"
  - «`Registrarse`» («Otros»)
  - Resolved tantos como sea posible en esta sesión

```
#include <iostream>
using namespace std;
int factorial(int n) {…}
int main (void) {
        int numcasos, N;
        cin >> numcasos;
        for (int i= 0; i<numcasos; i++) {
                cin >> N;
                cout << factorial(N) << endl;
        }
}
```