Convenciones de programación que utilizaremos en IP

Comentarios:

```
/* Comentario de                    // Comentario de una sola línea
*  varias líneas
*/
```

Constantes: deben escribirse con todas las letras en mayúscula y las palabras separadas por un carácter de subrayado ("_").

```
const float PI = 3,14159;
const int MAX_ASISTENTES = 100;
```

Variables: seguir la convención *Lower CamelCase*: todas las letras en minúscula salvo la primera letra de cada palabra (excepto la primera), que se escribirá en mayúscula.

```
float notaMedia;
int numTotalAsistentes;
bool mayorQueLaMedia;
```

Tipos de datos no primitivos[1]: seguir la convención *Upper CamelCase*: todas las letras en minúscula salvo la primera letra de cada palabra (incluida la primera), que se escribirá en mayúscula (p.ej., `Persona`, `TableroDeJuego`).

```
Persona alicia, juan;
TableroDeJuego miTableroDeJuego;
```

Módulos (funciones y procedimientos): seguir la convención *Lower CamelCase*. Cada módulo realiza una tarea y, por lo tanto, los nombres de los módulos deben incluir un verbo en infinitivo que indique la acción que lleva a cabo. Así, también resulta más sencillo distinguir los nombres de los módulos de los de las variables.

```
int sumar ( int a, int b ) { ... }
float calcularMedia ( float a, float b ) { ... }
```

Convenciones especiales:

- Módulos que devuelven un valor Booleano: esXXXX

```
bool esPar ( int n ) { ... }
bool esDivisor ( int divisor, int numero ) { ... }
```

- Módulos que devuelven el valor de un miembro de una estructura de datos[2] (*getters*): obtenerXXXX o getXXXX

```
int obtenerEdad ( Persona p ) { ... }
```

- Módulos que permiten modificar el valor de un miembro de una estructura de datos[2] (*setters*): modificarXXXX o setXXXX

```
void modificarEdad ( Persona p, int nuevaEdad ) {...}
```

Bloques de código: delimitados con llaves y convenientemente tabulados.

```
if ( edad < 18 ) {
    ...
}
else {
    ...
}
```

```
if ( valor1 < 10 ) {
    ...
    if ( valor2 > 10 ) {
        ...
    }
}
```

```
while ( contador <= 100 ) {
    ...
}
```

```
int sumar ( int a, int b ) {
    return a + b;
}
```

---

[1] Veremos cómo extender el conjunto de tipos de datos primitivos de C++ más adelante en el curso.
[2] Presentaremos las estructuras de datos (`struct`) en C++ más adelante en el curso.