# Programación I

## 10 Buenas Practicas de Programación

*Universidad Tecnológica Nacional - Cuch Sede Chivilcoy*

*Profesores*

*Luciana Denicio*

*Matias Garro*

**1) Prioriza la legibilidad.** Aunque es natural tener que poner por delante la optimización, la legibilidad es mucho más importante. Debes escribir un tipo de código que cualquier desarrollador pueda comprender. Ten en cuenta que cuánto más complejo sea tu código, más tiempo y recursos serán necesarios para tratarlo.
2) Estructura la arquitectura. Una de las buenas prácticas para programadores más populares es estructurar una arquitectura concreta. Antes de dar el disparo de salida y escribir, piensa en la utilidad del código, en cómo funciona, como modula y con que servicios es compatible. Plantéate qué estructura tendrá, de qué forma lo testearás y cómo será actualizado.

3) Lee mucho código fuente. Aunque escribir código fuente es mucho más sencillo que entender el qué otros han escrito, es importante nutrirse del conocimiento ajeno. Si te esfuerzas en comprender el código de otros desarrolladores, podrás comprobar en un instante las diferencias entre código de calidad y código mediocre.
4) Coloca comentarios. Si te encuentras en una fase de aprendizaje, lo mejor es que coloques comentarios en tu propio código. Así, evitarás desorientarte cuando leas las funciones más complejas que tú mismo has creado. Además, si un tercero tiene que acceder a tu código, los comentarios le facilitarán la tarea.

5)Testea tu código. Indiferentemente de la longitud del código que hayas escrito, debes testearlo para comprobar que todo esté bien. Recuerda que encontrar un error a tiempo y solucionarlo evitará problemas en el futuro. Por ejemplo, los test son especialmente necesarios cuando se escribe código open source.

6) Simplifica al máximo. Trata de evitar la construcción de código complejo siempre que sea posible. Así, encontrarás menos bugs y ahorrarás tiempo en solucionar errores. Tu objetivo debería ser el de escribir código funcional, sin filigranas.

7) Realiza control de versiones. Utiliza algún software de control de versiones para gestionar los cambios que se apliquen sobre los elementos del código. De esta manera podrás conocer en qué estado se encontraba el código antes y después de ser modificado. Algunos ejemplos son Git o Subversion, fundamentales para evitar errores graves.

8) No reproduzcas fragmentos idénticos de código. Aunque hayas ideado un código estable y robusto, no debes copiar y pegar fragmentos para aprovecharlos en otros módulos. En su lugar, trata de encapsular esta parte del código en una función y aprovecharla cuando sea necesario.

9)Evita los elementos no habituales. Algunos lenguajes contienen elementos únicos distintos al resto. Es habitual que estos elementos sean utilizados por programadores de alto nivel, pero no están al alcance de todo el mundo. Evita estos elementos para que tu código no sea excesivamente críptico.

10) No utilices caracteres únicos del español. Ten en cuenta que caracteres como la «ñ» o las tildes generarán errores al no ser caracteres ASCII. Los archivos cuyo código contenga estos caracteres no recomendados podrían sufrir alteraciones al abrirse en diferentes equipos. Por ello, es recomendable que escribas código en inglés.