**¿Cómo aprender a Programar?**

\*Pasos para ser programador autodidacta:

1. Mentalizar que quiero se programador. Querer hacer esto, tener determinación.
2. Estudiar: Roles, tecnologías, métodos, que se hace que no, que debo hacer. Comunicarse con la computadora. Saber que estoy haciendo. Qué tipo de programadores son requeridos. Tener idea de lo que quiere el mercado.
3. Que programador quiero ser. A qué tipo de programación quiero entrar.
4. Crear un portafolio. O sea, vender lo que se. Crear algo (páginas web, por ejemplo) y meterlo en un portafolio.
5. Entrevista: Aplicar a trabajos, ir a entrevistas, saber que responder, prepararme para una entrevista de trabajo.
6. Repite: Falla y aprende de los errores.
7. Trabaja: Ser un programador junior. Escoger un trabajo, cualquiera que sea y ganar experiencia.

\*Cómo aprender a programar rápido:

1) Enfocarse: Si pierdo tiempo, ver cómo puedo cambiar mis comportamientos. Pasar horas concentrado frente al monitor. Mente clara dentro de la programación.

2) La zona: Sentarse en frente de la computadora e involucrarse en lo que estoy haciendo. Concentrarse, ser claro, y no perder el hilo de lo que estoy haciendo.

3) Aprender conceptos, no solo el lenguaje: Aprender cómo funciona un lenguaje de programación. Para que sirve, que hace, como funciona. Entender cómo funcionan las variables, las estructuras de datos, los arrays y objetos, los árboles, condicionales, bucles o loops, ciclos, algoritmos, etc.

4) Menos cursos, más proyectos: Es mejor practicar para comprender que es lo que estoy haciendo y si lo estoy logrando.

5) Generar experiencia: La mejor forma es haciendo proyectos, como crear aplicaciones o páginas web.

6) Aprender como aprender: Saber cómo aprender, o sea, adaptarme a los cambios y las nuevas tecnologías, a saber, cómo explicar Python, o como funciona y saber exactamente qué es y para que lo uso.

\*Lógica de Programación:

1) Uso de la programación: La programación se utiliza con dos objetivos principales. El primero es manipular datos, por ejemplo, tener una lista de elementos, tomar esos datos y mostrarlos en pantalla, esto sería como procesar datos. También puedo hacer una “calculadora”, que sería tomar datos de números y hacer operaciones. El segundo uso es dar instrucciones a la computadora, o sea, dar instrucciones al hardware, como controlar periféricos, o que se prenda ciertas luces, o que la cámara se prenda, etc.

2) Variables: Son pequeños espacios de memoria donde se guardan datos. Los tipos de datos que pueden encontrarse son, textos, números decimales y enteros, booleanos, una lista con múltiples tipos de datos, diccionarios o mapas (mapear un valor a otro valor). Estos datos no son constantes, sino que pueden cambiar con el tiempo.

3) Constantes: Guarda los datos, pero no los permite cambiar. Es muy parecido a las variables.

4) Funciones: Aquí se guarda toda la lógica de programación, se utiliza para que podamos ingresar datos, y que estos puedan ser procesados y mostrados en pantalla. Es una lista de funciones.

5) Condicionales: Permite dar un flujo al procesamiento de datos. Aquí podemos poner que, si no se cumplen ciertos parámetros, el programa no continuara con los procesos.

6) Ciclos, bucles o loops: Permite procesar datos uno por uno, especialmente en listas.

7) Sintaxis: Es conveniente usar un solo lenguaje de programación, debido a la sintaxis, ya que cada lenguaje tiene sus propios métodos. Es la forma en la que escribo el código.

8) Tipeados o no tipeados: Los tipeados requieren el uso de más código para escribir, y nos permite tener menos errores, ya que es más específico.

9) Import y export: Es importar y exportar, se usa para tener más orden para crear programas, ya que se utilizan más archivos para guardar las líneas. De aquí se crea las librerías, que son muchos archivos escritos que se usan para el proyecto que se está creando.

\*Errores de Programadores Novatos:

1) Aprender un lenguaje antes que lo lógica: Esto nos permitirá aplicar toda la lógica a cualquier lenguaje, porque la lógica es universal.

2) Buscar resultados rápidos: Todo aprendizaje lleva su tiempo, si algo lleva más tiempo para el aprendizaje, no hay que alarmarse.

3) No programar todos los días: Practicar todos los días, o casi todos. Si practicó lograre aprender mejor. No ser un estudiante eterno.

4) Hacer chistes del punto y coma: Tener cuidado cuando uno programa, y prestar atención al editor.

5) Copypaste: Solucionar problemas copiando/pegando no es lo mejor.

6) No usar control de versiones: Recomendable para no perder las versiones más viejas y tener más confiabilidad a la hora de llevar el control de las versiones.

7) Preocuparte por la edad: No incide en nada el tema de la edad.

8) No planificar: Ser ordenado cuando uno hace un código o proyecto. Planificar el programa. No echar el código como loco.

9) Tener miedo a preguntar: Si algo no es obvio, o no quedo claro, es mejor preguntar antes de no saber de qué trataba el tema.

10) En mi maquina funciona: El que funciona un código en una máquina, no significa que en otra también, ya que cada máquina tiene su propio entorno. En ese caso, es mejor documentar todo.

11) Usar frameworks/librerías sin conocer el lenguaje: Antes de usar frameworks o librerías, debemos conocer el lenguaje o la teoría de esto antes de usar esto, para comprender el uso de esto.

12) Pelearse por tecnologías: Los lenguajes no son inseguros, ni uno es mejor que otro, cada uno tiene su propio uso.

13) Código sucio: No mezclar todo un código, tener declarado todo, y la mejor manera es hacerlo a través de diferentes archivos, o crear funciones para no escribir mucho código.

14) No saber trabajar con otros programadores: Se trabaja en equipo, hay que saber cooperar y adaptarse a las reglas y pautas del equipo.

15) No descansar correctamente: Es importante descansar, ya que no somos una máquina, sino un ser humano común y corriente. Descansar en cuerpo y alma es lo mejor. Para trabajar bien hay que descansar bien.