Consejos programación

- Hacerlo de manera **incremental**.
- Ir **poco a poco**. Lento pero seguro. Despacito con buena letra. Paso a paso
- Ir **probando** cada cosa nueva que implementes, para así si aparecen errores, determinar exactamente porque se debe ese error y en donde esta.
- Dividir el código en **partes** y hacer funciones. **Comentar** donde empieza cada sección y donde acaba y que hace. Divide y vencerás
- Saber bien cual es **problema** que tienes que resolver y **planificar** como quieres que sea la solución final. Si tienes eso, ya solo es programarlo
- Hacerlo primero de una manera **menos** sofisticada por ejemplo sin interfaz y después ponerlo en interfaz.
- Puedes hacer una versión **inicial** que funcione peor estructurada y después ir **refactorizando**
- Hacerlo con **cabeza**, pensando porque hago cada línea. Cuidado programación por **asedio.** No probar cosas al azar a ver si funciona
- Saber que **almacena** cada variable y poner correctamente el **nombre**. Definir un **protocolo** si tienes muchas variables
- Añadir **comentarios** por ejemplo para saber las llaves a quien corresponden, con la estructura donde inicia cada parte. Partes más complicadas una descripción
- **Probar** en programas a parte más sencillos la funcionalidad que quieres implementar en el programa final
- Realizar **copias** de seguridad. Control de versiones
- Tener una **idea** de cómo va a ser el código. Por ejemplo, hago una clase de esto. **Planificar** antes de empezar. No ir a lo loco
- **Reutilizar** código. No **reinventar** la rueda. Coger de internet o de tu biblioteca
- Partir de una versión inicial**. Buscar ejemplos similares** a lo que quieres hacer y a partir de ahí modificar
- De lo más **general a lo mas especifico**. Primero realizar la estructura mas general del programa y después implementar las partes más específicas.
- Ir **depurando** con print. Mira que tienen las variables. O depura correctamente
- No te tienes que **agobiar** ni pasarlo mal. Tienes que disfrutarlo y para ello programa despacio. Si te agobias, respira, cálmate, relájate.
- Los **errores** y excepciones son normales. No suele funcionar a la primera.
- Investigar por internet errores o **formas** de hacerlo
- **Variables** al principio
- Antes de teclear planea toda la **estructura** y **pseudocódigo** con el flujo para las funciones
- Fichero donde realizar **pruebas**, antes de ponerlo en el programa final hacer prueba
- No **repetir** código
- Hacer **test**
- **Identación**
- No hagas un caos insostenible, te vas a agobiar. Que sea **escalable.** Que si tienes que añadir algo sepas donde esta y **controles** que hace cada parte
- Mejor hacerlo **una** vez **bien** que 2 veces y gastes más tiempo

- Componentes **independientes** y separados. Trozos o porciones que abstraes como si fuesen funciones y como hacen sus operaciones te da igual. Ejemplo: conectar socket
- Como si no fuese para ti, fuese para **alguien mas** y lo tuviese que entender
- Poner **ejemplos en papel** de como quieres que funcione y partir de ahí comprender lo que se quiere hacer
- **Definir en texto** que es lo que debe hacer el código, un párrafo describiendo el algoritmo que se seguirá
- Saber **que esta haciendo** el programa para acotar el fallo. Saber que hay en cada variable
- **Acotar el fallo**. Reducir los candidatos que pueden provocar el fallo. Para ello ir imprimiendo cada paso.
- Si lo tienes que implementar para que se haga muchas veces y repita algo muchas veces, primero **pruébalo con una sola repetición**. Por ejemplo, análisis de partidos, en lugar de probarlo con todos los partidos, mientras lo realizas, hazlo que solo procese un partido
- **Hacer en papel** dibujos en sucio de lo que se quiere implementar.