1. FFT, en C++, Ari
   1. Comparar con la implementación de referencia
   2. Test bench para entregar
2. Programa principal, Joaco
   1. Funcionalidad esencial: sintetizar una pista a partir de un midi, cargar un midi, asignar cualquier instrumento a cualquier track configurando los parámetros de cada uno, mezclar distintos tracks para generar una pista, guardar la pista como audio, mostrar el espectrograma (esto último puede estar aparte).
   2. Ensayarlo con un fragmento “considerable” del segundo movimiento (adagio) del concierto de Aranjuez, de Joaquín Rodrigo. Entregar el WAV
3. Síntesis aditiva de sonidos
   1. Por lo menos cuatro instrumentos distintos. “Encontrar los parámetros óptimos”
   2. Proponer una parametrización que no sea la ADSR
   3. Implementar caída de los parciales según la nota, desviación aleatoria de la frecuencia de forma apropiada para que sea “realista”
   4. Definir rango de funcionamiento
4. Síntesis mediante modulación en frecuencia
   1. Sintetizar mediante FM un clarinete. Evaluar con un espectrograma
   2. Sintetizar una campana
   3. Sintetizar otros instrumentos de viento
5. Síntesis de sonidos mediante modelos físicos
   1. Para modelo original y modificado:
      1. Encontrar analíticamente H(z) y respuesta en frecuencia (el modificado no es LTI slds)
      2. Implementar. “Observar salidas relevantes”. Investigar la influencia de ruido normal vs uniforme.
      3. Graficar polos y ceros. ¿Varían con el tiempo?
   2. Para modelo original:
      1. Rango de valores para RL para que el sistema sea estable, a partir de la transferencia
      2. Implementar un filtro que emule la caja de resonancia de una guitarra
      3. Investigar una solución para un problema de error en la frecuencia
   3. Para modelo modificado:
      1. Fase del filtro en función de b
      2. Expresión de b en función de beta para obtener el valor exacto de la fundamental deseada
      3. Desventaja introducida por este filtro, limitaciones, en qué octavas conviene usarlo
   4. Paper de Karlplus-Strong: implementar para distorsión y feedback.
6. Síntesis basada en muestras
   1. Obtener varias muestras de uno o varios instrumentos e implementar un sintetizador basado en muestras
   2. Métodos PSOLA o WSOLA, otros
7. Espectrograma, Rochi
   1. Función que calcule y grafique el espectrograma de una señal, cambiando tamaño de bloque y tipo de ventana
   2. ¿Cuál es el overlap requerido en un espectrograma según la ventana empleada? Ahre
   3. Obtener espectrograma de una escala de G3 generada por síntesis aditiva con 120ms por nota. ¿Parámetros apropiados?
8. Efectos de audio, en C++ con PortAudio u otra librería cross platform
   1. Eco simple
   2. Reverberador plano
   3. Reverberador pasa-bajos
   4. Reverberador completo
   5. Reverberador por convolución
   6. Que el usuario pueda ajustar los coeficientes de los filtros
   7. Proveer presets
   8. Flanger: encontrar respuesta en frecuencia e implementar
   9. Robotización de voz
   10. Giro 3D con auriculares
   11. Vibrato
   12. Aplicar los efectos a wav (puede ser programa aparte)