**Project 1**

**Objectives:**

Project consists of writing software to produce the audio file.

After completing the Project you will be familiar with:

1. Creating, compiling and running C++ projects on Linux system
2. C++ arrays
3. Using functions in C++
4. Basics of Digital Signal Processing (DSP)

**To Submit:**

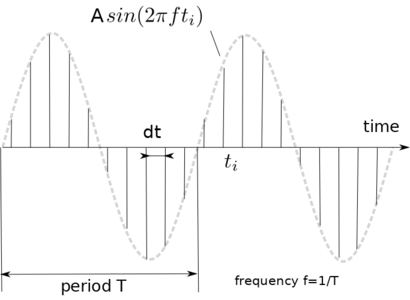
* Three C++ source files: submit one \*.cpp file for Core, one for Completion and one for Challenge.
* Your Reflection.pdf file - can be one file. Please label sections.

**Harmonic tone (Core) [35% for working code, 10% for reflection]**

**Code**

**Write program which can generate single-tone sound.**

Waveform of single-tone sound:



**1** Make new project

* Start Geany
* Create new file and save it with expension .cpp
* Write empty int main() function
* Compile and build the file to check that Geany is working

**2** Introduce variables and select their types

* Select sampling rate (samples per second). Think which type is better to use for this variable
* Calculate time interval between samples *dt.* Make variable.
* Select *duration* of the tone.
* Calculate number of samples *nSamples = duration / dt.*
* Reserve memory for array of **int** big enough to contain all samples
* Select frequency of the tone *f*

**3** Write the algorithm

* Calculate values of array elements to following formula:

 \begin{displaymath}a_i = A \cdot sin(2 \pi f t_i) \end{displaymath} 

where

 \begin{displaymath} t_{i} = i \cdot dt \end{displaymath} 

is current time for this sample

 \begin{displaymath} A \end{displaymath} 

is volume of the sound

* Calculations should be repeated n\_samples times. You can consider using **for()** operator
* Save **a** as *.wav* file. See Appendix for details.

**4** Test your program

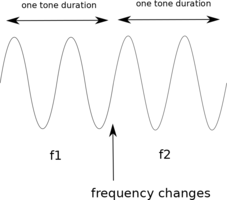
* Run your program
* Check that program does not crash. If it does you may want to add some output (**cout>>**) to your code to localize problem point
* Inspect produced file. If you saved file right it is possible to inspect sound waveform using **Sonic Visualiser**software, which you can find under Multimedia menu on our system.
* Play the audio. Please use headphones not to disturb others.

**Reflection**

* Explain your reasoning for selected sampling rate
* Explain how you calculated total number of samples
* Explain your reasoning for selection of types of all variables
* Include screenshot of the waveform obtained with SonicVisualizer. Does waveform like what you intended?

**Ambulance Sound (Core + Completion)[25% for working code, 10% for reflection]**

Write program which can generate ambulance siren sound - made of two frequencies repeating number of times.



**Code**

* Make new project. You can also save your previous project with new name and modify it.
* There is no fixed way to complete this task. You can take different approach not one described below. As long as it works, there is no reduction in marks.
* Select duration of one tone sounding.
* Select how many times two tones will be repeated.
* Now you calculate how many samples there will be in the waveform.
* Reserve memory for waveform samples.
* Run cycle for number of repeats.
* Inside run another cycle generating audio signal of certain frequency. After completing that - change the frequency and run another cycle.
* Save generated waveform as wav file.

**Reflection**

* Find and describe alternative way to decide when to change the frequency. No need to code, just describe an algorithm.

**Grand Piano (Challenge) [10% for working code, 10% for reflection]**

Write program which can:  
**1** Read music score sheet (greatly simplified) (about 2.5% with regard to the marks) File is of the following format:  
174.6  
174.6  
195.996  
... where number is frequency of the note in Hz. Number of lines equals number of notes in melody. We don't know in advance how many notes melody contains. It would be beneficial if your program can read file of any length.  
For example of test file see Attachments.

**2** Play the melody (about 2.5% of marks) You can choose any **tempo** you like (note duration) If amplitude of waveform is constant music will sound atrificial.

**3** Make your program produce sound like real musical instrument. (about 5% with regard of the marks) There is an artistic licence in this part. This part is hard. To make audio to sound somewhat similar to real instrument amplitude of the sound has to change while note is playing. It can follow:

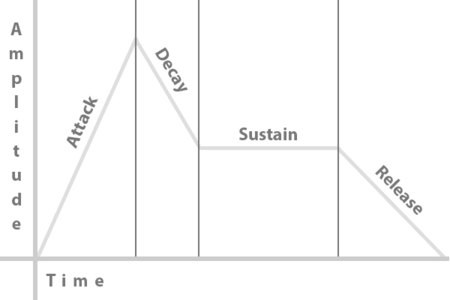


Image from: <https://www.teachmeaudio.com/recording/sound-reproduction/sound-envelopes>   
Piano, for example, will produce sharp "attack", no "sustain", long "decay", no "release". Trumpet - slow "attack", "sustain", no "decay", slow "release"

You may want to investigate how higher-order harmonics contribute to the sound quality too: <https://method-behind-the-music.com/mechanics/physics/>

**Reflection**

Explain your choice of note amplitude shape: attack, sustain, decay and release parameters.

**Appendix**

We provide helper software to make saving **.wa**