```
////////////////////////////////////       class_10_Base_Music         //////////////
////  this tab manage the function that govern the user about the base music

/// import the library that allow to play mp3 files
import ddf.minim.*;

/// this class manage the base music part of the software:
/// _plays mp3 files
/// _manage the passage between different music channel for each user
/// _set the correct chords for each choice
class Base_music{
  PApplet parent;
  Minim minim;

  AudioSnippet[] personChannels;

  int numPersonChannels;
  int currentPersonChannel = 0;
  int numChords;
  int fadeStepTime = 0;
  float gainUpValue = 0;
  float gainDownValue = 0;
  int fadeUpDelay;

  int[] jump;
  int[] jump_set;
  int[] jump_min;

  int chord1;
  int chord2;
  int chord3;
  int chordType;
  int countBase=0;

  Base_music() {

    numChords = 7;
    // make the Minim sound object
     minim =new Minim(parent);

    numPersonChannels = 2;
     personChannels =new AudioSnippet[numPersonChannels];

    fadeUpDelay = 500;
/// set different harominc jump to chose the next chord for each tone
///
     jump=new int[3];
     jump[0]  = 1;
     jump[1]  = 4;
    jump[2] = 6;

     jump_set=new int[4];
     jump_set[0]  = 1;
     jump_set[1]  = 3;
    jump_set[2] = 5;
    jump_set[3] = 6;

     jump_min=new int[4];
     jump_min[0]  = 1;
     jump_min[1]  = 2;
     jump_min[2]  = 4;
    jump_min[3] = 6;


  }


  void Play_Base_music() {
    if(countBase==0){
      personChannels[0] = Chords[1][0];
      personChannels[0].loop();
      countBase++;
    }
    else{
      updateChange();
    }
  }
```

```
/// this void manage the correct passage between chords: set the fade out and the fade in of the sound everytime the user change pc
  void updateChange() {
    if (millis() > fadeStepTime) {
       fadeStepTime =millis() + (fadeUpDelay/46);
      if (currentPersonChannel == 0) {
        if (personChannels[0]  !=null) {
          if (gainUpValue < 6) {
            gainUpValue++;
            personChannels[0].setGain(gainUpValue);
          }
        }
        if (personChannels[1]  !=null) {
          if (gainDownValue > -40) {
            gainDownValue--;
            personChannels[1].setGain(gainDownValue);
          }
         }
       }
      else if (currentPersonChannel == 1) {
        if (personChannels[1]  !=null) {
          if (gainUpValue < 6) {
            gainUpValue++;
            personChannels[1].setGain(gainUpValue);
          }
        }
        if (personChannels[0]  !=null) {
          if (gainDownValue > -40) {
            gainDownValue--;
            personChannels[0].setGain(gainDownValue);
          }
         }
      }
     }
  }

  void changeChord(int toneType) {
    if (currentPersonChannel == 0) {
      if (personChannels[0]  !=null) {
        gainDownValue = 6;
      }


      chord1 = chord1 + jump[int(random(0,2))];
      if (chord1>=7){
        chord1=chord1-7;

      }
      chord2 = chord2 + jump_min[int(random(0,3))];
      if (chord2>=7){
        chord2=chord2-7;

      }
      chord3 = chord3 + jump_set[int(random(0,3))];
      if (chord3>=7){
        chord3=chord3-7;

      }

/// here the software load the correct mp3 file for each choice (for the channel 1)
      switch(toneType) {
      case  0:
        personChannels[1] = Chords[chord1][toneType];
        chordType = chord1;
        break;
      case  1:
        personChannels[1] = Chords[chord2][toneType];
        chordType = chord2;
        break;
      case  2:
        personChannels[1] = Chords[chord3][toneType];
        chordType = chord3;
        break;
      }

      gainUpValue = -40;
      personChannels[1].setGain(gainUpValue);
      personChannels[1].loop();
      currentPersonChannel = 1;
     }
    else if (currentPersonChannel == 1) {
      if (personChannels[1]  !=null) {
        gainDownValue = 6;
      }
      chord1 = chord1 + jump[int(random(0,2))];
      if (chord1>=7){
        chord1=chord1-7;
      }
      chord2 = chord2 + jump_min[int(random(0,3))];
      if (chord2>=7){
        chord2=chord2-7;
      }
      chord3 = chord3 + jump_set[int(random(0,3))];
      if (chord3>=7){
        chord3=chord3-7;
      }


/// here the software load the correct mp3 file for each choice (for the channel 2)
      switch(toneType) {
      case  0:
        personChannels[0] = Chords[chord1][toneType];
        chordType = chord1;
        break;
      case  1:
        personChannels[0] = Chords[chord2][toneType];
        chordType = chord2;
        break;
      case  2:
        personChannels[0] = Chords[chord3][toneType];
        chordType = chord3;
        break;
      }
      gainUpValue = -40;
      personChannels[0].setGain(gainUpValue);
      personChannels[0].loop();
      currentPersonChannel = 0;
    }
  }
  void stop(){
    minim.stop();
    PauseSound();
  }

  void PauseSound(){
    if (personChannels[0]  !=null) {
      personChannels[0].pause();
      personChannels[0].rewind();
    }
    if (personChannels[1]  !=null) {
      personChannels[1].pause();
      personChannels[1].rewind();
    }

  }

}
```