一.程序功能介绍:

实现对本地音乐的播放等功能,主要包括:添加本地音乐至曲库,将曲库中音乐自定义组合成播放列表，删除曲库、播放列表里的歌曲，播放音乐的自由选择,上一首、下一首，播放进度\音量\播放模式和倍速的调节,对曲库中音乐的搜索,歌词显示与滚动等。

二.项目各模块与类设计细节

曲库和播放列表的相关功能:

1.添加音乐：用户选择文件夹路径,将其记录,使用QDir将后缀为.mp3的文件路径读入,最终记录到QList<QUrl>SongLib中,并将文件名打印至List\_Lib(QListWidget).

2.搜索：先读入Search\_Lib(QLineEdit)中输入的内容,至输入Enter开始搜索。提取SongLib中的所有歌曲的文件名，使用contains函数等去除Search\_Lib前后的空格并进行模糊匹配,然后清空List\_Lib并将匹配到的结果打印到List\_Lib中，如果输入空白搜索，则打印所有歌曲。

3.双击List\_Lib中的Item移除指定曲目：获取指定的索引,从SongLib和List\_Lib删除。单击将指定曲目加入播放列表,将索引对应QUrl加入QList<QUrl>playlist,将其打印至List\_ToPlay(QListWidget) 。为了实现双击时不触发单击的函数，加入了一个计时器，单击时触发计时器，计时器计时到200ms时触发单击函数（即将指定曲目加入播放列表）。若在200ms内第二次点击触发双击函数，则会停止计时，防止触发单击函数

4.清空曲库：使用pushButton和clear函数清空List\_Lib和SongLib

播放相关:

1.播放与暂停。使用了player(QMediaPlayer\*)指针来指定播放位置,播放函数play():如果player非空,则调用play()函数,否则调用playCurrent()函数将player指向指定曲目并使用setSource函数和play()函数播放指定曲目。暂停则是直接使用pause()函数实现。以上操作均会触发Button\_PlayPause(QWidget)的图标切换。直接点击播放列表中的曲目时,playSelectedURL()函数将获得其索引,然后直接播放选定的曲目。

2.播放顺序选择,顺序\随机\单曲循环可选，改变的是player指针的变换方式，顺序加减1，随机random，单曲循环不变。

3.下一首,上一首。依据选择的播放顺序,通过不同的方式改变player指针。并且在播完一首时，可以自动播放下一首。

4.速率\音量,分别通过setVolume()和setPlaybackRate()进行调节。同时,点击音量键将直接setVolume(0.0)或setVolume()为记录的值，从而达到切换静音的目的。通过Button\_Mode的点击操作进行隐藏hide()和显示show()倍速Box的操作。

5.进度条。使用connect将音频进度的改变和音频时长的变化与Slider\_Process(QSlider)绑定,并将Slider\_Process的改变与音频进度绑定。

歌词相关:

1. 获取歌词：在通过点击播放歌曲时，程序会获取歌曲的地址，然后把歌曲地址的最后三个字符替换成lrc（例：Sample.mp3->Sample.lrc），这样就得到了歌词lrc文件的地址。然后用QFile函数打开文件，并用QString获取全部的歌词，关闭文件。然后新建一个class LyricLine，包含一行歌词的时间和内容两个信息。最后使用正则表达式对形如[mm:ss:xx]的时间信息进行遍历，从而记录每一行的歌词信息，并添加到一个名为Lyric的QList<LyricLing>中，这样就获取了歌词的时间和内容信息，并且按行分开。
2. 显示滚动歌词：通过QMediaPlay::PositonChanged函数实时获取歌曲播放情况，然后在updatelyrics函数中，先判断歌曲有没有歌词，如果没有就显示“纯音乐”，如果有，就把当前播放时间与Lyric中的时间进行匹配，从而确定当前歌词。然后用HTMl来高亮打印当前歌词，其余歌词普通打印。最后通过scrolltoAnchor函数，把歌词滚动到当前歌词的位置（如果歌词是前三行，就滚动到第一行，如果超过前三行，就滚动到当前歌词前的第三行，从而防止当前歌词显示在列表顶部，不美观）

美观相关：

1. 播放/暂停通过setIcon(QIcon())函数来改变图案。上一首、下一首图案通过在UI设计界面改变样式表的方式来设置图案（好处是方便，坏处是不能修改图标）。音量键在UI设计界面打开checkable，然后设置打开和关闭两种形态的图标，也可以实现图案的转变，并且再静音时还会高亮为蓝色
2. 布局广泛使用水平布局和垂直布局，并且设置了窗口的最小大小，多数控件的大小，从而保证窗口在放大或缩小时保持比例不变，保持美观
3. 曲库和播放列表，通过在UI界面的样式表中输入改变外观的代码，修改了列表的字体大小、行间距、背景颜色、selected时的颜色（去除了选取时的虚线框），鼠标滑过时的颜色等方式进行美化，并且播放列表的字体大小用的是原本的大小，以示区分
4. 界面图标修改成了音乐图标

三、小组分工情况：

刘智琦（组长）：

模糊搜索（升级自王石的搜索）+单击添加歌曲、双击删除歌曲和计时器+歌词的读取和滚动播放+外观设计+部分小功能和优化（一键静音、list的一键清除等）+组织分工

王石（组员）：

歌曲搜索+歌曲路径添加（曲库、播放列表）+排序+提供歌曲的resource+互助合作

楼宇乾（组员）：

基础音乐播放功能（播放暂停+下/上一首+模式切换（随机、顺序、循环）+音量、速率调节、自动播放下一首、进度条）

四、项目总结和反思：

通过本次项目，我们不仅增强了对Qt框架的熟练度，还提升了团队协作和解决问题的能力。在未来的学习和工作中，我们将继续积极探索新技术，不断提升自身的程序设计水平。

在本次项目中，我们从QQ音乐上获取灵感，决定做一个简易的音乐播放器，主要攻克乐曲的读取、载入以及一些基本的音乐播放的功能。我们在制定好分工的情况下，先自主开展自己的工作，最后在进行合并。但在合并时，我们会发现两个人之间的程序可能不能完美的合并起来，遇到了一些麻烦，比如思想的碰撞、音乐无法播放、进度条与播放未同步的情况。在查阅了很多资料的情况下，我们逐渐克服了这些困难，完成了功能可操作的音乐播放器。但我们并不满足于此，决定对此进行进一步的优化，增加了自动播放下一首、模糊排序、清空删除、歌词滚动等功能，不断精益求精，力争完善。

最后，我们成功地完成了一个基于Qt的音乐播放器项目，并取得了以下成果：

1、功能完善：实现了基本的播放、暂停、停止、上一曲、下一曲、音量速率调整、模式切换等功能，用户操作流畅。

2、界面美观：设计了更好的用户界面，包括直观的控制按钮、清晰的音乐信息显示和美观的背景。

3、播放列表管理：支持动态添加和删除歌曲，用户可以方便地管理自己的播放列表。

4、文件操作稳定性：确保音乐文件的加载和保存操作稳定可靠，可从自己的文件夹中获得音乐，操作简单。

5、歌词的可读性：添加了歌词的滚动操作，让人知道歌曲内容，更易于学习跟唱。

当然在项目开发过程中，我们遇到了一些挑战和需要改进的地方：

1、 需求管理与变更控制

在项目初期，我们没有充分预见到需求变更的可能性，导致后期频繁调整和修改功能。这给项目进度和团队的开发效率带来了一定的影响。未来在类似项目中，我们应该：

（1）加强需求分析：在项目启动阶段，与成员进行充分的需求沟通和确认，确保初步需求尽可能完整和准确。

（2）灵活应对变更：实施更有效的变更管理措施，包括评估变更对项目的影响、优先级和成本。

2. 代码质量与优化

部分功能实现时的算法和代码结构未经过充分优化，这影响了程序的性能和可维护性。为了改进这一点，我们可以：

（1）优化和重构：定期检查和优化项目中的关键代码段，包括性能瓶颈和复杂度较高的部分。确保代码的清晰度和可维护性，减少后续维护的成本和风险。

（2）完善与更新：随着时代人们的需求变化，不断进行代码功能的调整与完善，以更好的迎合人们的需求以及使用。

4. 团队协作与沟通

尽管整体团队合作良好，但在沟通和协作方面还有改进的空间。我们可以：

（1）加强沟通渠道：通过更频繁的会议、团队讨论或使用协作工具，确保所有成员对项目目标和进展的理解一致，以及使用列表、函数、变量的可连接性和可替换性。

（2）进行知识分享：促进团队成员之间的技术交流和知识分享，不断互帮互助，以提升整体团队的能力和效率。

在这场程序设计的头脑风暴中，我们逐渐从个人走向整体，从残破走向晚上，我们不仅收获了QT程序设计的知识，并加以实践，更是收获了简易音乐播放器的设计成就，为未来打下良好的编程基础。