1. 程序功能介绍：
利用 QT 实现桌面插件项目，在主界面直接显示北京的实时天气信息，并且可以搜索国内主要城市获取实时天气信息。通过菜单按钮实现更改壁纸，增加桌面贴图，显示时钟，倒计时，番茄钟功能。点击日历会出现萌宠，双击萌宠过后进行日历与事件备忘功能。
2. 模块实现：
（1）贴图类 attach：
通过更改样式表消除边框，设置透明度，增设背景图面，并与鼠标点击信号槽关联，
在构造函数中即随机生成了贴图
on_closeButton_clicked()，
on_minButton_clicked()，
mouseMoveEvent(QMouseEvent* ev)，
void mousePressEvent(QMouseEvent* ev)与信号关联
通过初始化设置消除边框，并且使背景透明
（2）desktopbk：桌面背景类
成员函数 setsedesktopimage()
生成随机数，在构造好的壁纸图片库以数字.jpg 形式命名方便更改壁纸
通过调用 windows API 更改系统设置 SystemParametersInfoA(SPI_SETDESKWALLPAPER, 0, byte.data(), SPIF_UPDATEINIFILE | SPIF_SENDWININICHANGE);
（3）weatherdata 模块:
class Now：用于记录从 api 获取的当前北京市天气信息
成员变量：
city 记录城市名称
weatherstate:记录天气状态
code：对应的天气图片的编码
tempera：记录当前温度
在后续的使用过程中，访问网址
https://api.seniverse.com/v3/weather/now.json?key=S1lY6DHhdSxqo396N&location=%1&language=zh-Hans&unit=c 调用 API 获取天气信息后 json 解压保存到 mnow 这一对象中，
通过 updateui 函数设置 icon 与 text 以及字体大小等进行实时更新

（4）倒计时 Qt_countdown：
构造函数和析构函数：
Widget(QWidget *parent)：构造函数，初始化 UI 组件，设置窗口属性（如不透明度、背景色和无边框），初始化定时器并连接定时器的超时信号到槽函数 do_timer_timeout，设置窗口置顶。
~Widget()：析构函数，释放 UI 资源。
事件处理函数
mousePressEvent(QMouseEvent *e)：鼠标单击事件处理函数，当左键按下时，记录单击位置。
mouseMoveEvent(QMouseEvent *e)：鼠标移动事件处理函数，当左键按下并移动时，窗口跟随鼠标移动。
功能函数
subtractOneSecond()：倒计时减少一秒的逻辑函数，处理时间的减少，如果时间减至零，返回 false。

do_timer_timeout()：定时器超时处理函数，调用 subtractOneSecond 减少一秒并更新显示，若时间减至零，停止定时器。
on_btnStart_clicked()：开始按钮点击处理函数，启动定时器并从 timeEdit 组件获取初始时间，更新 hms 变量并在控制台打印小时、分钟和秒数。
on_btnStop_clicked()：停止按钮点击处理函数，停止定时器。
（5）番茄钟 Qt_tomatoClock：
类：TomatoClock
成员变量
Opacity：窗口的不透明度，初始值为 0.8。
state：表示当前状态（1 表示运行，0 表示暂停）。
tomato_num：记录番茄钟的次数。
current_color：当前背景色，初始值为浅红色#CD9B9B。
timer：QTimer 类型的定时器指针，用于控制时间更新。
time：QTime 类型的时间变量，用于存储和更新计时。
构造函数和析构函数
TomatoClock(QWidget *parent)：构造函数，初始化 UI 组件，设置窗口属性（如不透明度、背景色和无边框），初始化定时器，连接定时器的超时信号到槽函数 updateTime，设置窗口置顶并启动定时器。
~TomatoClock()：析构函数，释放 UI 资源。

事件处理函数
mousePressEvent(QMouseEvent *e)：鼠标单击事件处理函数，当左键按下时，记录单击位置并根据当前状态暂停或恢复计时。
mouseMoveEvent(QMouseEvent *e)：鼠标移动事件处理函数，当左键按下并移动时，窗口跟随鼠标移动。
功能函数：
initTime()：初始化时间，将时间复位为 0 并更新 LCD 显示。
updateTime()：更新时间函数，每秒增加一秒，更新 LCD 显示，根据时间改变背景色和番茄钟状态。
（6）动态时钟 Qt_Clock：
类：User
成员变量
Opacity：窗口的不透明度，初始值为 0.8。
current_color：当前背景色，初始值为浅红色#CD9B9B。
mouse_flag：鼠标按下标志。
mouse_x、mouse_y：鼠标按下时的坐标。
m_x、m_y：窗口大小。
in_style：时钟风格选择标志。
m_clickClose：右键点击次数，用于关闭时钟。
m_clickColor：左键点击次数，用于切换时钟颜色。
UI 组件：包含标题、文本、按钮等。
构造函数和析构函数
User(QWidget *parent)：构造函数，初始化 UI 组件，设置窗口属性（如无边框、透明度、背景色等），并连接信号与槽函数。

~User()：析构函数，释放资源。

事件处理函数

mousePressEvent(QMouseEvent *ev)：鼠标按下事件处理函数，记录鼠标位置，并根据鼠标按键进行处理（左键更换颜色，右键关闭窗口）。

mouseReleaseEvent(QMouseEvent *ev)：鼠标释放事件处理函数，重置鼠标标志。

mouseMoveEvent(QMouseEvent *ev)：鼠标移动事件处理函数，实现无边框窗口的拖动功能。

槽函数

begin()：隐藏当前窗口。

big()：设置窗口大小为大。

medium()：设置窗口大小为中。

small()：设置窗口大小为小。

style_s()：选择简单风格。

style_p()：选择多边形风格。

功能函数

Text(QString text)：返回介绍文本内容。

类：AnalogClock

头文件

analogclock.h：包含 AnalogClock 类的声明。

成员变量

m_user：指向 User 类的指针。

m_timer：QTimer 类型的定时器指针，用于刷新时钟。

mouse_flag：鼠标按下标志。

mouse_x、mouse_y：鼠标按下时的坐标。

m_clickClose：右键点击次数，用于关闭时钟。

m_clickColor：左键点击次数，用于切换时钟颜色。

hourColor、minuteColor、secondColor：时针、分针、秒针的颜色。

构造函数和析构函数

AnalogClock(QWidget *parent)：构造函数，初始化 UI 组件和定时器，设置窗口属性（如无边框、透明度等），并连接信号与槽函数。

~AnalogClock()：析构函数，释放资源。

事件处理函数

mousePressEvent(QMouseEvent *ev)：鼠标按下事件处理函数，记录鼠标位置，并根据鼠标按键进行处理（左键更换颜色，右键关闭窗口）。

mouseReleaseEvent(QMouseEvent *ev)：鼠标释放事件处理函数，重置鼠标标志。

mouseMoveEvent(QMouseEvent *ev)：鼠标移动事件处理函数，实现无边框窗口的拖动功能。

绘图函数

paintEvent(QPaintEvent *event)：绘制时钟，包括时针、分针、秒针和时钟刻度等。

多边形绘制函数

Polygon(int edge, int r, double P_x[], double P_y[])：绘制正多边形。

Polygon2(int edge, int r, double P_x[], double P_y[])：绘制起始点不在 y 轴上的正多边形。

（7）日历相关：

首先具有主窗口 BackGround 类，用于实现电脑桌面装饰和界面跳转功能

类内具体构造如下：
覆写 mouseDoubleClickEvent(QMouseEvent *event)函数，实现双击跳转
覆写 eventFilter(QObject* watched, QEvent* ev)函数，读取鼠标位置并实现拖动效果
具有一个 QLabel* rolelabel，用于展示界面图片
在构造函数中不断更新状态，去掉界面空白窗口并定时调用 updateRoleAnimation()函数更新动画
MainMzk 类：
MainMzk 窗口由 BackGround 窗口跳转，继承自 QDiolog，为日历的核心显示界面类.
类内具体构造如下：
存储了一个 std::unordered_set<Date,std::hash<Date>> DateList;
元素是一个自定义的 Date 类（Date=每天的日程需要显示的信息+一个 hash 函数）
有 ModifyDate(const Date&),覆写了 closeEvent(QCloseEvent *event)函数，围绕 Datelist 展开，确保日期的正确读取，写入和保存。
Date 类：
Date 类继承于 QDate，一方面是为了解决 QDate 无法放入无序哈希表的缺点，构造了一个 QDate 的哈希函数，从而使得 Date 可以被存取到 unordered_set 中；另一方面是为了存取每日的日程安排，利用向量 std::vector<Event> ToDos 来存取.
哈希函数采用整数的哈希函数作用在一个关于日期唯一的特征数上
此特征数是：372 * (year()-2000) + 31 * month() + day()
这样的选取可以保证很长时间的日期在进入整数的哈希函数前不重复
Event 类包含一个日程所需的各种信息：

```
int sTime[2];//起始时间，分别是小时和分钟
int eTime[2];//终止时间
QString _ToDoText;//需要做的事情
QString _Address;//事情的位置
char Color[3];//事件颜色
```

Calendar 类：
在双击日期后跳出 QDateModift 类，继承自 QDiolog，表现为右侧的边栏
由一个滚动区域 scrollArea 和一个[新建日程]的按钮组成
scrollArea 用于滚动显示 EventShow 类，即这样的小框 

其需要的各种信息 QDateModift 从父类 MainMzk 中获得的

Eventshow 类
内置了双击的判断，采取了另一种时间测定和记录鼠标点击次数的方法.
在点击日程左侧的按钮之后，日程会变成已完成状态，表现为字符上添加一条删除线.如右图所示
在填写时间时，在双击后会使文本编辑框 QLineEdit 显示在最上方，在填写之后利用对应的函数反馈到显示窗口，从而实现信息的更改，同时内置了对时间的合法性判断和缺失的 0 填充.如 12:9 会被识别为 12:09 并填入
（8）mainwindow：
通过 push_button 按钮，槽函数与之前的类相关联

3. 具体分工：

路景轶：完成对时钟、番茄钟、倒计时功能的实现，

王路阳：完成日历与备忘录日程的实现，

朱家庆：完成更改壁纸、设置桌面贴图、主界面天气预报的功能实现，整合文件

4. 总结与反思：

目前实现了桌面组件的一些常用基本功能，实用性较强。

但功能可以更加完善，比如完善更加详细的天气预报信息，从实时信息推广到未来多日的信息，也不局限于仅仅是天气，还可以加入更多空气质量，日相月相等信息。以及加入闹钟语音提示功能，让日程信息显示的更加清晰。