结束python语言初步学习后，进入Qt编程学习。学习中在很多地方直接使用他人已经定义好的类，因此安装了很多第三方模块，在安装与实践中不断尝试解决进而更深入地理解路径、系统。使用类的过程中，老师提出不必过度纠结一些代码原理，善于使用帮助手册辅助学习的观点。首先创建QApplication，再使用from、import引入类，其中QtWidgets，QtCore，QtGui，QThread是其中重点类。在这一过程中老师穿插了面向对象的代码复用方式——继承。Qt的高级编程涉及人机交互，信号和槽的理解有一定难度。Qt应用模版更是简单易理解，既实现数据封装代码高级，又方便用户使用。当同一项目中包含多个界面且需依次调用摄像头时，需要设置信号并绑定槽，借助release()函数在线程结束时及时释放摄像头设备。

安装opencv相关插件后在老师指引下查询官方网站提供的帮助手册学习了几个重要函数和类，进一步学习图像与界面相关知识。其中core：图像的基本表示，imgproc：图像处理，imgcodecs：图像解码：图像IO，videoio：摄像头与视频的IO。图像采用cv2.imread（）加载图像，返回numpy.ndarray图像类型，以数组形式存储，彩色图片是一个(height，width，channel/depth)的三维数组，深度在opencv中关联通道和通道的基本类型。由此可知道构造一个numpy.ndarray就是构造一个图像，借助python的数据可以相当方便的实现图像构造，而对数组尤其是数组下标操作可以较好的实现图像处理，而下标操作又可分整数、切片、逻辑矩阵三大类，考虑到时间有限，老师只能在诸多的处理中选择部分经典处理进行演示，但足以令人惊叹python语言的方便与简洁，在课后的练习中逐步了解图像处理，但函数较多时常记不清具体对应功能。结束静态图像处理后，开始学习动态视频下的数据抓取与图像处理。

从静态图像处理到动态视频处理，先使用Qt的designer工具设计ui组件界面，再使用pyuic5 -o将ui翻译成python后通过self.ui.setupUi()使用已创建的ui对象，其中需要使用import引入相关的python模块。之后继承QThread借助override run函数创建摄像头对象，实现视频采集与数据抓取，此前模糊的线程的意义的功能在这一课有了进一步的理解。定义合适的信号和槽函数可以传递图片并在Qt中显示，处理图片的方式与之前相似。为了更好的图像识别和图像特征分析，老师还补充了卷积的相关知识，并介绍了jupyter工具推导数学原理公式，不由得感慨数学知识的重要性。

理解图像与数组的关系后，开始深入学习torch工具，从构造张量到操作张量，再设置函数求导与运算，其中穿插了gpu与cpu的对比。之后借助torch中的数据集以及相关函数操作，以手写数字识别为例，教授torch与opencv的格式转换，并演示通过简单的卷积运算和线性运算将一张图像转换为多个特征并计算概率。概率的计算依赖相关模型，而模型训练依赖数据集及相关训练。数据集的建立需要较多的照片，借助精灵标注助手标注图像后生成需进行格式转换的文件，通过借助python对相关文件配置调整即可建立自己的数据集。在最开始时我们直接使用已写好的YOLO进行训练并测试验证，后续收集好相关图像后开始使用自己的数据集训练。