西北农林科技大学信息工程学院
面向对象程序设计实习报告

题　目：LaTeX科技排版实习报告

学　　号	2019012971
姓　　名	何志杰
专业班级	软件工程1903班
指导教师	魏蕾
实践日期	2020年8月24日—9月4日

目　　录

摘　　要

针对现实生活中人工登记学生信息繁琐、数据易丢失等问题，我们基于Qt5制作了有图形化GUI界面的学生信息管理系统，具有易登记、易保存信息等特点。本系统基于面向对象程序设计，实现了基本的学生信息增删改查功能，还额外添加了学生学号、成绩排序等功能。数据的存储采用文件来实现，可移植性能强。

关键词: 学生信息管理系统; Qt5程序设计; C++; 面向对象程序设计

一、 综合训练目的与要求

该综合训练的目的是培养应用面向对象设计方法及解决实际问题的能力，掌握使用LaTeX科技排版技术, Qt5开发，熟悉开发工具的使用。同时提升调查研究、查阅文献及编写技术文档的能力。

二、 综合训练任务

采用面向对象思想设计stuinfo类，数据结构采用链表node类来实现，通过对链表的修改以达到增、删、改、查、排序功能的实现。此外对于使用程序的不同用户，设计不同的入口让用户进入不同的界面。

三、 总体设计

(1) 程序运行架构

对于我们设计的基于QT5框架下的GUI程序，我们将学生的相关信息存储在stuinfo.txt文件中，由于没有使用数据库相关技术，在登录界面上只能使用提前设置好的账号密码验证管理员身份(在本程序设计中, 管理员账号为admin, 密码为123456)，对于学生用户，在登录界面可以直接选择学生界面，同样可以查找和排序(查看) 的操作，从而实现不同用户的对程序的使用。如图 1所示：

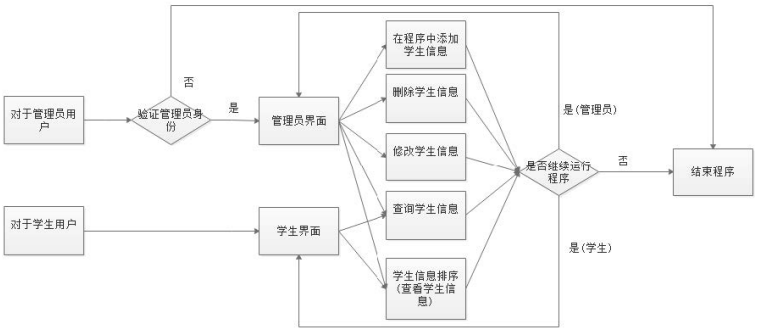


图1 程序整体架构图

(2) 代码文件组成

该程序代码为标准的QT5 Widgets Application 的工程代码，包含以下几部分，如表 1所示：

表1 程序的代码组成

文件名	后缀	实现的功能
addstudentwidget	.cpp .h .ui	增加学生信息界面
adminmenuwidget	.cpp .h .ui	管理员模式菜单界面
adminwidget	.cpp .h .ui	管理员模式窗口界面
delwidget	.cpp .h .ui	删除学生信息界面
findwidget	.cpp .h .ui	查找学生信息界面
mainwidget	.cpp .h .ui	登录主窗口程序界面
sortwidget	.cpp .h .ui	排序（输出）学生信息菜单界面
stumenuwidget	.cpp .h .ui	学生模式菜单界面
stuwidget	.cpp .h .ui	学生模式窗口界面
node	.cpp .h	以链表形式存储信息和功能实现函数
stuinfo	.cpp .h	以类表示学生信息
main	.cpp	主函数
stuinfo	.pro QT5	工程文件

(3) 函数列表

对 addstudentwidget.cpp，包含构造和析构函数，以及返回主菜单按钮的槽函数on_returnButton_clicked、增加学生信息的槽函数on_addButton_clicked。

对adminwidget.cpp，包含构造和析构函数。

对adminmenuwidget.cpp，包含构造和析构函数，以及各个按钮的槽函数。

对 delwidget.cpp，包含构造和析构函数，以及返回主菜单按钮的槽函数on_returnButton_clicked、删除学生信息的槽函数on_deletepushButton_clicked。

对 findwidget.cpp，包含构造和析构函数，以及返回主菜单按钮的槽函数on_returnButton_clicked、查找学生信息的槽函数on_pushButton_clicked。

对mainwidget.cpp，包含构造和析构函数。

对menuwidget.cpp，包含构造和析构函数，以及登录界面各个按钮的槽函数。

对 modifywidget.cpp，包含构造和析构函数，以及返回主菜单按钮的槽函数on_returnButton_clicked、查询修改前学生信息的槽函数 on_searchButton_clicked，修改学生信息的槽函数on_modifypushButton_clicked。

对 sortwidget.cpp，包含构造和析构函数，以及返回主菜单按钮的槽函数on_returnButton_clicked、获取学生信息的函数 getStuInfo，std::sort 所需的排序函数

cmp_number 和cmp_score，排序学生信息的槽函数on_modifypushButton_clicked。
对stuwidget.cpp，包含构造和析构函数。
对stumenuwidget.cpp，包含构造和析构函数，以及各个按钮的槽函数。
对stuinfo.cpp, 包含构造和析构函数，以及传递信息的相关函数，在此不表。
对Node.cpp, 包含构造函数，从文件获取信息函数InputStudent，删除学生信息函数DeleteStudent，输出信息到文件函数OutputStudent, 修改学生信息函数 ChangeStudent，查找学生信息函数SearchStudent, 相关函数的声明如下。

- void InputStudent();
- void DeleteStudent(long long number);
- void OutputStudent();
- void ChangeStudent(QString name,long long number,QString age,QString gender, long long tel,QString bir,QString address,double score);
- bool SearchStudent(QString &name,long long number,QString &age,QString &gender, long long &tel,QString &bir,QString &address,double &score);

四、 详细设计说明

(1) 增加学生信息函数

这是一个无参函数，实现新增一位学生信息的功能，新的学生信息节点默认添加到链表尾部。算法：定义节点指针p，指向第一个学生信息节点p=pHead->pNext，通过 while 循环将其指向尾节点，读入学生信息并将其写入新建节点pt 中，p->pNext 指向新节点pt，pt->pNext 为NULL，添加完成。流程图如图 2:

(2) 输入学生信息函数

Void InputStudent() 这是一个无参函数，实现将文件中存储的学生信息读入到程序中。算法：先声明一个头节点pHead，并创建一个指向头节点的指针pTail，每输入一个学生的信息就声明一个新节点pNew 来存储信息，把pNew 挂到老节点后面，同时移动pTail 到pNew 上并使pTail->pNext = NULL。流程图如图 3：

(3) 输出学生信息函数

这是一个无参函数，负责对链表中全部学生信息的输出。算法：先将p 节点的指针指向第一个节点，将p 节点（即第一个节点）的数据输出。然后再将p 节点的指针指向下一个节点，即p = p -> pNext，再次将 p 节点的数据输出。重复执行此步骤直到p 指针指向NULL 为止。流程图如图 4：

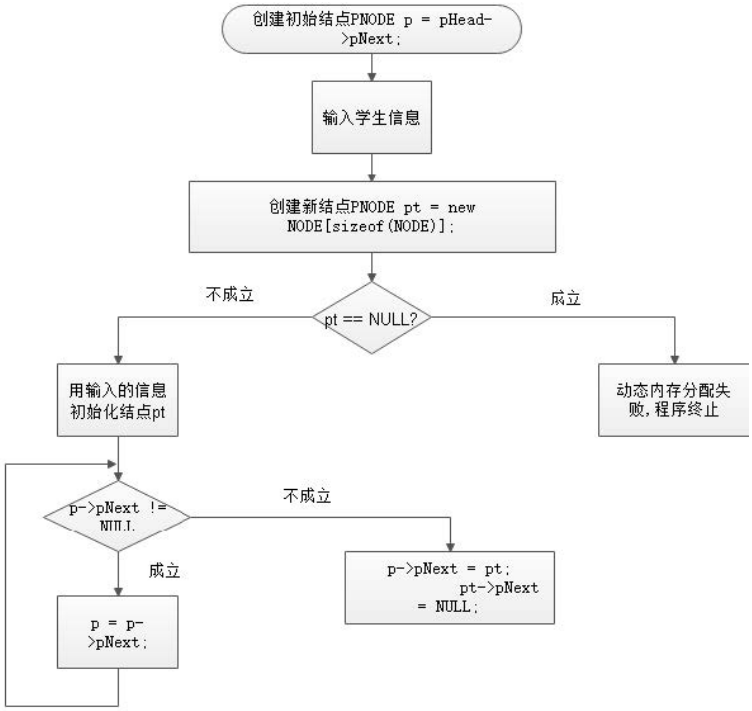


图2 增加学生信息函数流程图

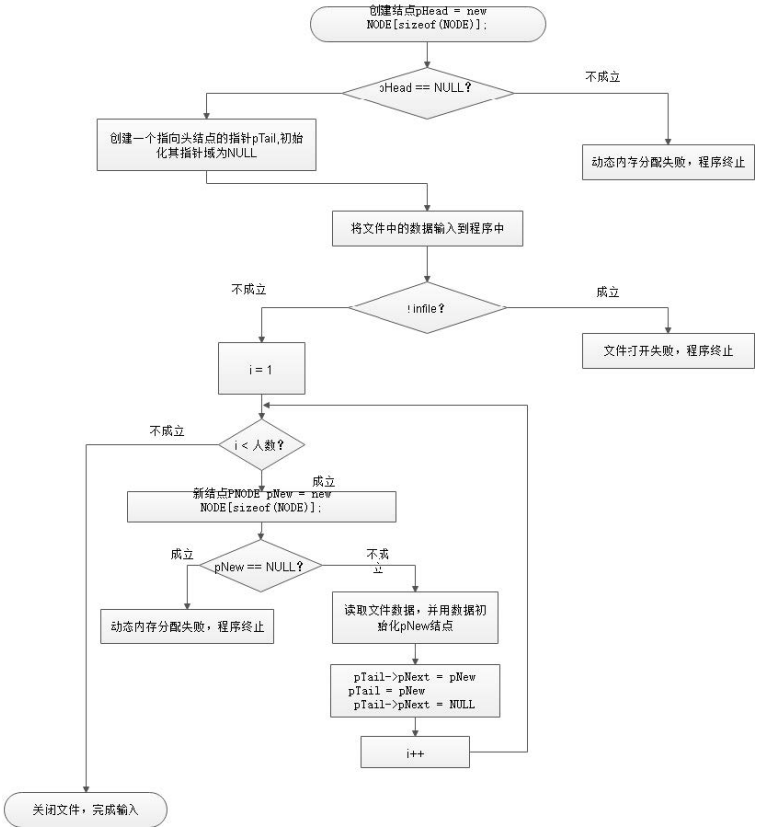


图3 输入学生信息函数流程图

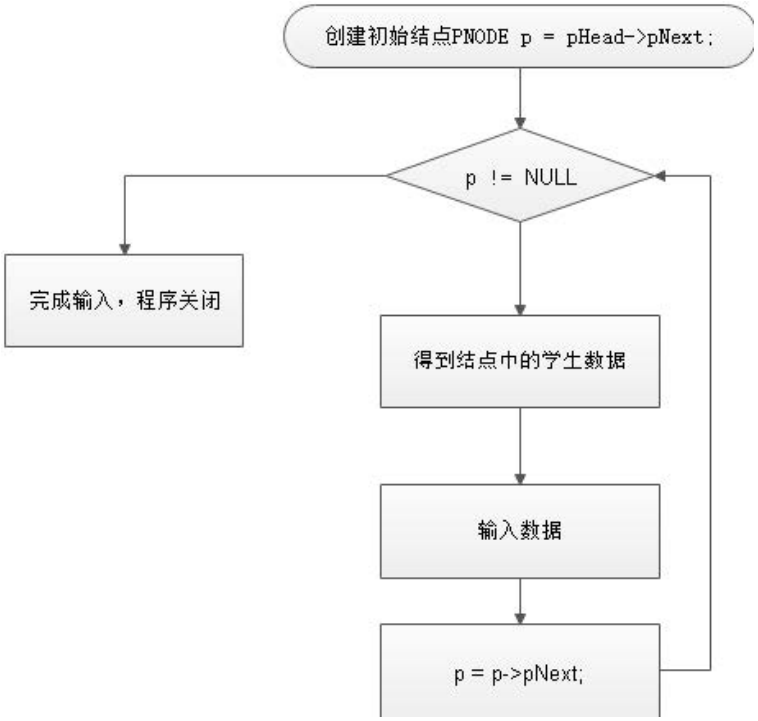
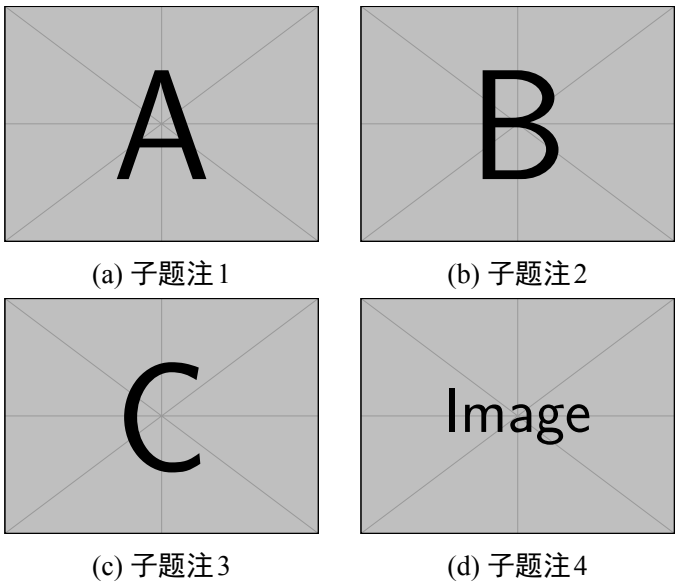


图4 输出学生信息函数流程图



(a) 子题注1　　(b) 子题注2

(c) 子题注3　　(d) 子题注4

图5 四个子图

五、 temp

在模板中，已为插图设置了./figs/、./figure/、./figures/、./image/、./images/、./graphics/、./graphic/、./pictures/、./picture/相对路径，可以在当前工作路径中任意一个这样命名的文件夹，以存放需要的插图文件。

(1) 表格浮动体

如果需要插入一个简单的表格，可以仅使用table和tabular环境实现，如表 2。

注意:科技文档中的表格需要采用**三线表**。

表2 城市人口数量排名 (source: Wikipedia)

城市	人口
Mexico City	20,116,842
Shanghai	19,210,000
Peking	15,796,450
Istanbul	14,160,467

如果多个表格布局较为复杂，可以使用 floatrow 宏包实现排版。如用代码 1可编制表 3和表 4所示的横向并排表格。

代码清单1: 表格排版

```
% 横向排版两个表格
\begin{table}[!htp]
  \begin{floatrow}
    \ttabbox[\FBwidth]
    {
      \begin{tabular}{ccl}
        \toprule
        序号 & 测试用例 & \multicolumn{1}{c}{测试目的}\\
        \midrule
        1 & a & 小写字母到大写字母转换\\
        2 & A & 大写字母到小写字母转换\\
        3 & @ & 非字母字符测试\\
        4 & 2 & 非字母其它字符\\
        \bottomrule
      \end{tabular}
    }{\caption{字母大小写转换测试用例表}\label{tab:testsample}}
    \ttabbox[\FBwidth]
    {
      \begin{tabular}{lr}
        \toprule
        城市 & 人口 \\
        \midrule
```

```
            Mexico City & 20,116,842\\
            Shanghai & 19,210,000\\
            Peking & 15,796,450\\
            Istanbul & 14,160,467\\
            \bottomrule
          \end{tabular}
        }{\caption{城市人口数量排名}\label{tab:city2}}
      \end{floatrow}
    \end{table}
```

表3 字母大小写转换测试用例表

序号	测试用例	测试目的
1	a	小写字母到大写字母转换
2	A	大写字母到小写字母转换
3	@	非字母字符测试
4	2	非字母其它字符

表4 城市人口数量排名

城市	人口
Mexico City	20,116,842
Shanghai	19,210,000
Peking	15,796,450
Istanbul	14,160,467

如果表格内容很多，导致无法放在一页内的话，需要用longtable 或longtabu 进行分页。表 5是一个长表格的例子。

表5 实验测试数据

测试程序	正常运行时间(s)	同步时间(s)	检查点时间(s)	卷回恢复时间(s)	进程迁移时间(s)	检查点文件(KB)
CG.A.2	23.05	0.002	0.116	0.035	0.589	32491
CG.A.4	15.06	0.003	0.067	0.021	0.351	18211
CG.A.8	13.38	0.004	0.072	0.023	0.210	9890
CG.B.2	867.45	0.002	0.864	0.232	3.256	228562
CG.B.4	501.61	0.003	0.438	0.136	2.075	123862
CG.B.8	384.65	0.004	0.457	0.108	1.235	63777
MG.A.2	112.27	0.002	0.846	0.237	3.930	236473
MG.A.4	59.84	0.003	0.442	0.128	2.070	123875
MG.A.8	31.38	0.003	0.476	0.114	1.041	60627
MG.B.2	526.28	0.002	0.821	0.238	4.176	236635
MG.B.4	280.11	0.003	0.432	0.130	1.706	123793
MG.B.8	148.29	0.003	0.442	0.116	0.893	60600
LU.A.2	2116.54	0.002	0.110	0.030	0.532	28754

续下页

续表 5　实验测试数据

测试程序	正常运行时间(s)	同步时间(s)	检查点时间(s)	卷回恢复时间(s)	进程迁移时间(s)	检查点文件(KB)
LU.A.4	1102.50	0.002	0.069	0.017	0.255	14915
LU.A.8	574.47	0.003	0.067	0.016	0.192	8655
LU.B.2	9712.87	0.002	0.357	0.104	1.734	101975
LU.B.4	4757.80	0.003	0.190	0.056	0.808	53522
LU.B.8	2444.05	0.004	0.222	0.057	0.548	30134
CG.B.2	867.45	0.002	0.864	0.232	3.256	228562
CG.B.4	501.61	0.003	0.438	0.136	2.075	123862
CG.B.8	384.65	0.004	0.457	0.108	1.235	63777
MG.A.2	112.27	0.002	0.846	0.237	3.930	236473
MG.A.4	59.84	0.003	0.442	0.128	2.070	123875
MG.A.8	31.38	0.003	0.476	0.114	1.041	60627
MG.B.2	526.28	0.002	0.821	0.238	4.176	236635
MG.B.4	280.11	0.003	0.432	0.130	1.706	123793
MG.B.8	148.29	0.003	0.442	0.116	0.893	60600
LU.A.2	2116.54	0.002	0.110	0.030	0.532	28754
LU.A.4	1102.50	0.002	0.069	0.017	0.255	14915
LU.A.8	574.47	0.003	0.067	0.016	0.192	8655
LU.B.2	9712.87	0.002	0.357	0.104	1.734	101975
LU.B.4	4757.80	0.003	0.190	0.056	0.808	53522
LU.B.8	2444.05	0.004	0.222	0.057	0.548	30134
EP.A.2	123.81	0.002	0.010	0.003	0.074	1834
EP.A.4	61.92	0.003	0.011	0.004	0.073	1743
EP.A.8	31.06	0.004	0.017	0.005	0.073	1661
EP.B.2	495.49	0.001	0.009	0.003	0.196	2011
EP.B.4	247.69	0.002	0.012	0.004	0.122	1663
EP.B.8	126.74	0.003	0.017	0.005	0.083	1656

(2) 插图标注

在“nwafuprojrep.cls”模板中，可能通过引入了改自tikz-imagelabels宏包的tikz-imglabels宏包，利用TikZ为插图进行标注。该宏包的使用细节与 tikz-imagelabels完全一致，请在命

令行使用“texdoc tikz-imagelabels”命令查看其使用说明书。代码 2用于实现图 6所示的插图标注。

代码清单2: 插图标注代码

```
\begin{figure}[!htp]
  \centering
    \begin{annotationimage}{width=0.8\textwidth}{figs/01reviewicons01}
      % 绘制外观设置按钮分组示意下划线
      \draw[thick,blue] (0.86,0.26) -- (1.0,0.26);
      % 添加各图标标注
      \foreach \ann/\xpos in
      {
        {附\\注\\工\\具}/0.02, {高\\亮\\工\\具}/0.07,
        {下\\划\\线\\工\\具}/0.126, {删\\除\\线\\工\\具}/0.18,
        {删\\除\\线\\并\\插\\入\\附\\注\\工\\具}/0.229, {插\\入\\文\\本\\工\\具}/0.283,
        {文\\本\\工\\具}/0.346, {文\\本\\框\\工\\具}/0.40,
        {铅\\笔\\绘\\图\\工\\具}/0.45, {铅\\笔\\擦\\工\\具}/0.51,
        {图\\章\\工\\具}/0.56, {附\\加\\文\\件\\工\\具}/0.63,
        {绘\\图\\工\\具}/0.7, {保\\持\\选\\择\\工\\具}/0.79,
        {注\\释\\外\\观\\设\\置}/0.93
      }
      {
        \draw[annotation below = {{\ann} at \xpos}] to (\xpos,0.48);
      }
    \end{annotationimage}
    \caption{插图标注}\label{fig:annot}
  \end{figure}
```

(3) 流程图

在“nwafuprojrep.cls”模板中，可能引入了自己开发的tikz-flowchart宏包进行流程图的绘制。请在Github平台查看tikz-flowchart宏包的使用说明书。

在绘制流程图时，可以先用纸和笔绘制草稿，然后根据草稿布置各个结点，再连接流程线。这样，可以做到心中有数，绘制较为方便。

图 7a是一个流程图草稿。根据该草稿，基于tikz-flowchart宏包，用代码 3可绘制图 7b所示的流程图。

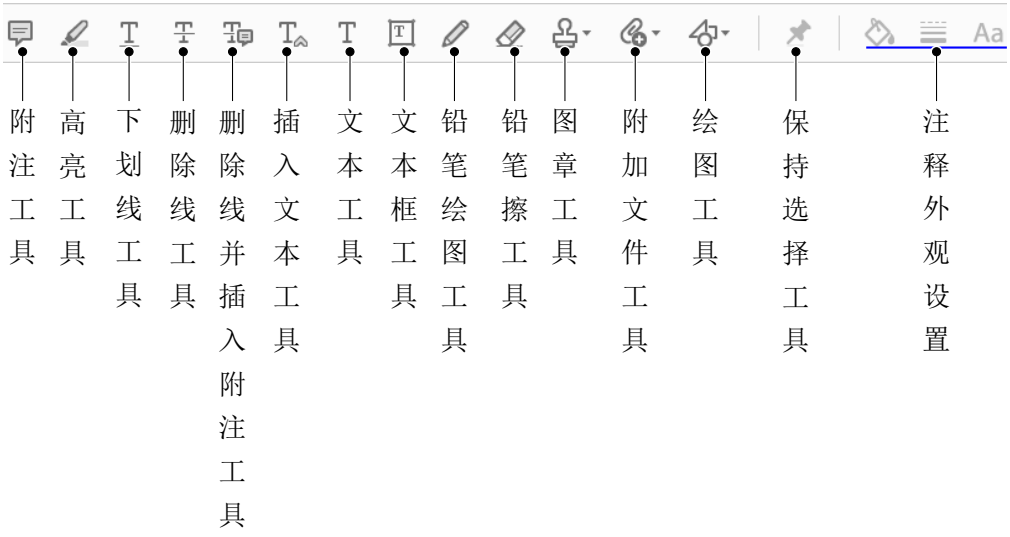


图6 插图标注

代码清单3：绘制流程图

```
% 流程图绘制属性设置
\flowchartset{
  proc fill color = orange!10, % 顺序处理框填充颜色(默认取白色)
  test fill color = green!30, % 判断框填充颜色(默认取白色)
  io fill color = blue!30, % 输入/输出框填充颜色(默认取白色)
  term fill color = red!30, % 开始/结束框填充颜色(默认取白色)
}
% 绘制流程图
\begin{tikzpicture}[scale=0.53,transform shape,]
  % 布置结点单元
  \node [term] (st) {开始};
  \node [proc, text width = 5em, join] (p1) {int divisor};
  \node [test, join] (t1) {n <= 1};
  \node [proc, text width = 5em] (p2) {divisor = 2};
  \node [test, text width = 10em, join] (t2) {divisor * divisor
      <= n};
  \node [test, text width = 8em] (t3) {n \% divisor == 0};
  \node [proc, text width = 5em] (p3) {divisor++};
  \node [term, below = 1.6 of p3] (end) {结束};
  \node [proc, text width = 4em, left = 4.8 of t2] (p4) {return
      0};
  \node [proc, text width = 4em, right = 3.5 of p3] (p5) {
     return 0};
  \node [proc, text width = 4em, right = 5.8 of t3] (p6) {
     return 1};

  % 布置用于连接的坐标结点，同时为其布置调试标记点。
  \node [coord] (c1) at ($(p2.south)!0.5!(t2.north)$)  {}; \
     cmark{1}
```

```
25        \node [coord, below = 0.25 of p3] (c2)  {}; \cmark{2}
26        \node [coord, above = 0.5 of end] (c3) {};  \cmark{3}
27        \node [coord, left = 0.5 of t2] (ct) {};  \cmark{t}
28        \node [coord] (c4) at (c3 -| p5)  {}; \cmark{4}
29        \node [coord] (c5) at (c2 -| ct)  {}; \cmark{5}
30
31        % 判断框连线，每次绘制时，先绘制一个带有一个固定
32        % 位置标注的路径(path)，然后再绘制箭头本身(arrow)。
33        \path (t1.south) -- node [near start, right] {$N$} (p2.north)
              ;
34        \draw [norm] (t1.south) -- (p2.north);
35        \path (t1.west) -| node [near start, above] {$Y$} (p4.north);
36        \draw [norm] (t1.west) -| (p4.north);
37
38        \path (t2.south) -- node [near start, right] {$Y$} (t3.north)
              ;
39        \draw [norm] (t2.south) -- (t3.north);
40        \path (t2.east) -| node [near start, above] {$N$} (p6.north);
41        \draw [norm] (t2.east) -| (p6.north);
42
43        \path (t3.south) -- node [near start, right] {$N$} (p3.north)
              ;
44        \draw [norm] (t3.south) -- (p3.north);
45        \path (t3.east) -| node [near start, above] {$Y$} (p5.north);
46        \draw [norm] (t3.east) -| (p5.north);
47
48        % 其它连线
49        \draw [norm](p3.south) |- (c5) |- (c1);
50        \draw [norm](p4.south) |- (c3);
51        \draw [norm](p4.south) |- (c3) -- (end);
52        \draw [norm](p5.south) -- (c4);
53        \draw [norm](p6.south) |- (c3);
54        \draw [norm](p6.south) |- (c3) -- (end);
55    \end{tikzpicture}
```

(4) UML图

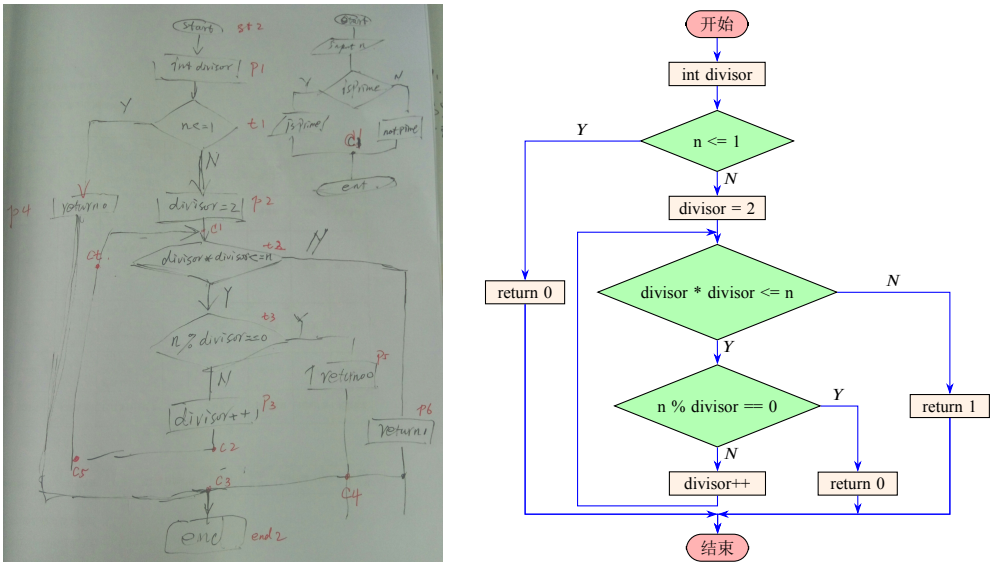
可以使用“pgf-umlcd”宏包实现UML图的绘制，在命令行使用“texdoc pgf-umlcd”查看该宏包的使用说明书。

但由于TeXLive收集的pgf-umlcd宏包不能很好的处理中文类名，因此请使用模板提供的pgf-umlcd宏包排版UML图。

图 8是使用“pgf-umlcd”宏包绘制的UML图。

(5) 代码排版

在“nwafuprojrep.cls”模板中，可以引入了自己开发的boxie.sty宏包进行代码排版。本文档第 九 节中有基本使用样例，详情请在Github平台查看 boxie宏包 的使用说明书。



(a) 草图　　(b) TiKZ绘制图

图7　用TiKZ绘制流程图

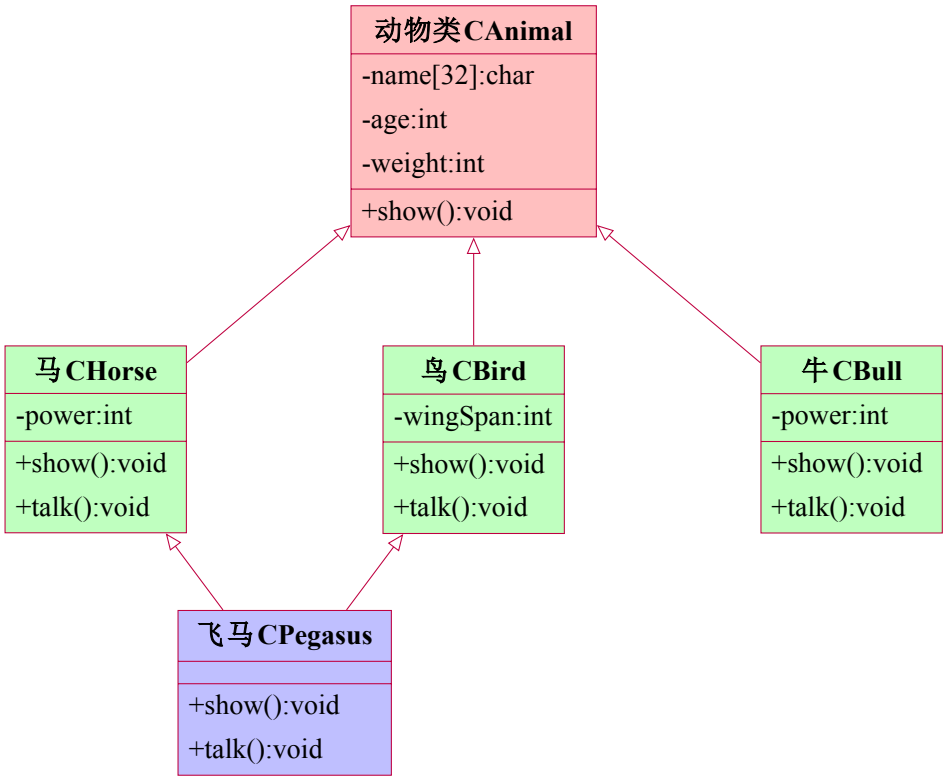


图8　用“pgf-umlcd”宏包绘制UML图

(6) 列表环境

在“nwafuprojrep.cls”模板中，基于enumitem宏包分别对itemize、enumerate和description三个环境的各个距离参数进行了修正，以使其排版结果符合中文习惯的首先缩进格式。

① itemize环境

- 床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光。
- 疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜。
- 举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月。
- 低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡。

② enumerate环境

1) 床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光。

2) 疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜。

3) 举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月。

4) 低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡。

③ description环境

床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光，床前明月光。

疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜，疑是地上霜。

举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月，举头望明月。

低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡，低头思故乡。

(7) “emph”强调字体

在“nwafuprojrep.cls”模板中，重定义强调字体，将默认强调字体是italic，中文用楷体代替操作更换为加粗操作，用加粗后的字体表示强调。

(8) 文本框盒子

文本框盒子继承于自己开发的boxie宏包，其使用细节请在Github平台查看 boxie宏包的使用说明书。同时，在该宏包的基础上，为boxie宏包添加加了摘自于 progartcn 论文模板的“标题”、“注意”、“重要”、“技巧”和“警告”文本框环境代码[1]。

① “标题”文本框

标题文本框环境的使用格式为：

```
\begin{titledBox}{<title>} <content> \end{titledBox}
```

HTTP/Console 内核

HTTP 内核继承自 Illuminate\Foundation\Http\Kernel 类，该类定义了一个 bootstrappers 数组，这个数组中的类在请求被执行前运行，这些 bootstrappers 配置了错误处理、日志、检测应用环境以及其它在请求被处理前需要执行的任务。

② “注意”文本框

注意文本框环境的使用格式为：

```
\begin{noteBox} <content> \end{noteBox}
```

注意

HTTP 内核继承自 Illuminate\Foundation\Http\Kernel 类，该类定义了一个 bootstrappers 数组，这个数组中的类在请求被执行前运行，这些 bootstrappers 配置了错误处理、日志、检测应用环境以及其它在请求被处理前需要执行的任务。

③ “重要”文本框

重要文本框环境的使用格式为：

```
\begin{importantBox} <content> \end{importantBox}
```

重要

HTTP 内核继承自 Illuminate\Foundation\Http\Kernel 类，该类定义了一个 bootstrappers 数组，这个数组中的类在请求被执行前运行，这些 bootstrappers 配置了错误处理、日志、检测应用环境以及其它在请求被处理前需要执行的任务。

④ “技巧”文本框

技巧文本框环境的使用格式为：

[1]本节示例摘自于该模板中的tutorial-sample.tex文件

\begin{tipBox} <content> \end{tipBox}

技巧

HTTP 内核继承自 Illuminate\Foundation\Http\Kernel 类，该类定义了一个 bootstrappers 数组，这个数组中的类在请求被执行前运行，这些 bootstrappers 配置了错误处理、日志、检测应用环境以及其它在请求被处理前需要执行的任务。

⑤ **“警告”文本框**

警告文本框环境的使用格式为：

\begin{warningBox} <content> \end{warningBox}

警告

HTTP 内核继承自 Illuminate\Foundation\Http\Kernel 类，该类定义了一个 bootstrappers 数组，这个数组中的类在请求被执行前运行，这些 bootstrappers 配置了错误处理、日志、检测应用环境以及其它在请求被处理前需要执行的任务。

(9) 交叉引用

在技术文档中，图、表、公式等必须使用**交叉引用**，在“nwafuprojrep.cls”模板中，交叉引用用\autoref命令实现引用，并对图、表、节、小节、公式、代码等引用标记字/词进行了设置。如对一个标签为“fig:01”的图使用 \autoref{fig:01}便可以得到“图 XX”的结果，用 \autoref{texcode01}就可以得到“代码 XX”的结果。

(10) 参考文献

参考文献采用GB/T7714-2015标准的顺序编码制，使用biber+biblatex方式实现，样式控制选择“biblatex-gb7714-2015”宏包的顺序编码制 (gb7714-2015)样式，如代码 4：

代码清单 4：引用参考文献

```
1  %% 引用参考文献
2  详见文献\cite{Peebles2001-100-100}\parencite{Babu2014--}
3  另见文献\cite[49]{于潇2012-1518-1523}\parencite[106]{Babu
       2014--}
```

能够生成如下引用结果：

详见文献[1][2] 另见文献[3]49[2]106

(11) 已载入的宏包

在“nwafuprojrep.cls”模板中，已引入的宏包有：etoolbox、unicode-math、geometry、ulem、array、graphicx、fancyhdr、titletoc、caption、footmisc、url、enumitem、circledsteps、filehook，无需再次引入这些宏包。

(12) 重要文件

重要

在使用在“nwafuprojrep.cls”模板前请确保：nwafuprojrep.cls模板文件在当前工作文件夹中。

如果需要代码盒子等各类盒子排版，则需要确保boxie.sty、fvextra.sty、lstlinebgrd.sty 这3个宏包文件在当前工作文件夹中，并且需要引入boxie宏包，boxie宏包会根据需要加载fvextra和lstlinebgrd 宏包，这两个宏包无需手动加载。

如果需要绘制UML图，则需要加载pgf-umlcd宏包，此时需要确保pgf-umlcd.sty宏包文件在当前工作文件夹中(TeXLive自带的宏包无法正确处理中文类名称，同时，修改了其association命令，以符合龚晓庆教材绘图习惯)。

如果需要绘制流程图，则需要加载tikz-flowchart宏包，此时需要确保tikz-flowchart.sty 宏包文件在当前工作文件夹中。

如果需要为插图进行村注，则需要加载tikz-imglabels宏包，此时需要确保tikz-imglabels.sty 宏包文件在当前工作文件夹中(TeXLive自带的宏包无法正确标注中的换行问题)。

Makefile文件是执行make命令需要的脚本文件，可以根据需要选择。

.latexmkrc文件是执行latexmk命令需要的脚本文件，可以根据需要选择。

(13) 排版编译

重要

由于需要排版代码文件，建议使用minted宏包实现代码排版，因此请确保安装有Python及其Pygments模块，详情请使用texdoc minted命令查看 minted宏包的使用说明。同时，为支持minted编译，请使用-shell-escape编译参数。

由于需要交叉引用和参考文献，建议按如下方式执行4次编译：

① 使用 minted 宏包排版代码

4 次 编 译

```
xelatex -shell-escape main.tex
biber main
xelatex -shell-escape main.tex
xelatex -shell-escape main.tex
```

如果使用 Makefile，则执行 make 命令即可：

4 次 编 译

```
make
```

如果使用 .latexmkrc，则执行 latexmk 命令即可：

4 次 编 译

```
latexmk
```

② 使用 listings 宏包排版代码

4 次 编 译

```
xelatex main.tex
biber main
xelatex main.tex
xelatex main.tex
```

如果使用 TeXstudio 等 IDE 工具，请参阅相关资料对 IDE 进行必要的配置。

建议通过 王泽鹏老师的 B 站资源，观看耿楠制作的教学视频学习必要的编译和配置过程。

六、 调试与测试

该模板是第 1 次发布，仅进行了部分测试，还需要在使用中不断完善。如果发现问题，请及时在 gitee 中提交 issues，以便改进代码。

七、 实习日志

实习日志用自定义的“logentry”环境编排，该环境第 1 个参数是日志标题该参数为可选参数，第 2 个参数是日志日期，该参数为必选参数，日期输入格式务必是“yyyy/mm/dd”格式，**注意**：星期会自动计算，无需输入。

2020 年 6 月 17 日星期三：会议讨论

开会讨论，决定使用 LaTeX 为实习报告排版提供支持，需要开发 nwafuprojrep.cls 模板。

2020 年 6 月 28 日星期日：完成开发

完成开发nwafuprojrep.cls模板v1.0.0开发，并进行了简单测试。在gitee平台开源发布nwafuprojrep v1.0.0模板。

2020年6月29日星期一：修正无Python的Pygments下代码排版问题

在无Python的Pygments模块下，使用listings宏包实现代码排版，并在Windows平台下完成了测试，同时，发布nwafuprojrep模板v1.0.1版。

2020年6月30日星期二：添加“logentry”日志环境

为实习日志编写添加自定义“logentry”环境，该环境设计了2个参数，第1个参数是可选参数，用于排版日志标题；第2个参数是必选参数，用于排版日志日期。

同时，对“main.tex”说明文档进行了完善和修定，添加了UML图的绘制示例。

将pgf-umlcd、floatrow、subcaptio、boxie、tikz-imglabels、tikz-flowchart宏包移出cls模板文件，以减少模板文档类的依赖，提高其灵活性。

如果需要，需将这些宏包置于当前工作目录下，建议在settings/packages.tex文件中引用这引宏包，在settings/format.tex中进行“设置floatrow浮动体属性”、“代码交叉引用命令autoref的引用格式”等设置。当然，这些引用和设置也可以在导言区完成。

同时，发布nwafuprojrep模板v1.0.2版。

2020年7月1日星期三：添加操作系统及字体判断功能

借鉴zepinglee开发的中国科学技术大学学位论文LaTeX模板代码, 为\nwafuset命令添加了操作系统和字体选择key-value选项，并实现了操作系统自动判断及字体自动选择功能。

2020年7月2日星期四：简化模板宏包依赖

进一步简化了nwafuprojrep.cls中对其他宏包的依赖，仅添加必要和宏包。其他宏包由用户决定是否需要加载，建议按推荐的目录结构在settings/packages.tex中添加需要的宏包。

添加报告(report)/论文(paper)选项，添加打印稿(print)/电子稿选项(epub)。

添加摘要排版命令，实现摘要排版。

修订README.md说明文件。

2020年7月6日星期一：调整模板宏包

在nwafuprojrep.cls中添加ulem、calc和array宏包，以使单独使用 nwafuprojrep.cls模板时，也可以正常编译，同时删除 settings/packages.tex中ulem和calc的宏包。

删除封面中“题目”、“学号”和“姓名”间自动添加空格计算，减少对calc宏包的依赖。

在.gitignore排除对main.pdf文件的跟踪，以减少仓库大小。

修订README.md说明文件，说明仅支持LaTeX 2020以后的发行版。

2020年7月10日星期五：调整实习日志日期输出格式

调整logentry实习日志环境的日期输出格式为中文格式，同时将日期输入格式调整为：yyyy/mm/dd。

八、 实习总结

通过本次实习，解决了广大学子在撰写实习报告时的排版问题，减轻了排版工作量，排版结果更标准、更专业.....

九、 附录：核心代码清单

可以使用boxie宏包提供的langPyOne、langCVOne等环境或langPyfile和langCVfile 等命令实现嵌入式代码或来自文件代码的排版。Py系列环境和命令不带引用计数和标签，CV系列环境和命令带有引用计数和标签。

(1) 代码排版环境

例如，使用langPyOne环境可以排版不带引用计数和标签的代码。

基于范围的循环

```
#include <iostream>
#include <vector>
using namespace std;
int main()
{
    // 累加 20 以内的素数
    int sum = 0;
    for(int e : {2, 3, 5, 7, 11, 13, 17, 19}) // 用 auto 类型更合理
    {
        sum += e;
    }
    cout << sum << endl;

    // 输出结果 77
    int arr[] = {1, 3, 5, 7, 9};
    // 声明数组 arr,初始化为 5 个奇数
    for(auto &ele : arr)
    {
        // 声明 ele,与数组 arr 关联在一起,用了 auto
        ele = ele * 2;
        // 修改数组每个元素的值
        cout << ele << " ";
        // 输出 ele,2 6 10 14 18
    }
    cout << endl;

    for(auto ele : arr)
    {
        cout << ele << " ";
    }
    // 没有改变:1 3 5 7 9
    cout << endl;

    return 0;
```

```
}
```

再如，使用langCVOne环境可以排版带有引用计数和标签的代码，如代码 5。

代码清单5: 基于范围的循环

```cpp
#include <iostream>
#include <vector>
using namespace std;
int main()
{
    // 累加 20 以内的素数
    int sum = 0;
    for(int e : {2, 3, 5, 7, 11, 13, 17, 19}) // 用 auto 类型更合理
    {
        sum += e;
    }
    cout << sum << endl;

    // 输出结果 77
    int arr[] = {1, 3, 5, 7, 9};
    // 声明数组 arr,初始化为 5 个奇数
    for(auto &ele : arr)
    {
        // 声明 ele,与数组 arr 关联在一起,用了 auto
        ele = ele * 2;
        // 修改数组每个元素的值
        cout << ele << " ";
        // 输出 ele,2 6 10 14 18
    }
    cout << endl;

    for(auto ele : arr)
    {
        cout << ele << " ";
    }
    // 没有改变:1 3 5 7 9
    cout << endl;

    return 0;
}
```

(2) 代码排版命令

代码排版命令用于根据代码文件进行排版，因此，可以在IDE中，例如 Code::Blocks中对代码进行排版，然后将排版后的文件直接用代码排版命令在 LaTeX中进行排版。

例如，使用langPyfile命令可以载入代码文件，排版不带引用计数和标签的代码 (注意指定必要的路径)。

基于范围的循环

```
#include <iostream>
#include <vector>
using namespace std;
int main()
{
    // 累加 20 以内的素数
    int sum = 0;
    for(int e : {2, 3, 5, 7, 11, 13, 17, 19}) // 用 auto 类型更合理
    {
        sum += e;
    }
    cout << sum << endl;

    // 输出结果 77
    int arr[] = {1, 3, 5, 7, 9};
    // 声明数组 arr,初始化为 5 个奇数
    for(auto &ele : arr)
    {
        // 声明 ele,与数组 arr 关联在一起,用了 auto
        ele = ele * 2;
        // 修改数组每个元素的值
        cout << ele << " ";
        // 输出 ele,2 6 10 14 18
    }
    cout << endl;

    for(auto ele : arr)
    {
        cout << ele << " ";
    }
    // 没有改变:1 3 5 7 9
    cout << endl;

    return 0;
}
```

再如，使用langCVfile命令可以载入代码文件，排版带引用计数和标签的代码 (注意指定必要的路径)，如代码 6。

代码清单 6: 基于范围的循环 C++

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 int main()
5 {
6     // 累加 20 以内的素数
7     int sum = 0;
```

```
    for(int e : {2, 3, 5, 7, 11, 13, 17, 19}) // 用 auto 类型更合理
    {
        sum += e;
    }
    cout << sum << endl;

    // 输出结果 77
    int arr[] = {1, 3, 5, 7, 9};
    // 声明数组 arr,初始化为 5 个奇数
    for(auto &ele : arr)
    {
        // 声明 ele,与数组 arr 关联在一起,用了 auto
        ele = ele * 2;
        // 修改数组每个元素的值
        cout << ele << " ";
        // 输出 ele,2 6 10 14 18
    }
    cout << endl;

    for(auto ele : arr)
    {
        cout << ele << " ";
    }
    // 没有改变:1 3 5 7 9
    cout << endl;

    return 0;
}
```

注：使用minted宏包排版代码，可以使用c++/C++或cpp指定语言名称，若使用listings宏包排版代码，则只能使用c++/C++指定语言名称

参考文献

[1] Von Peebles Jr., P. Z. Probability, random variable, and random signal Principles and LaTeX[M]. 4th ed. New York: McGraw-Hill, 2001: 100.

[2] Babu, B. V., NAGAR, A. K., DEEP, K., et al. Proceedings of the second international conference on soft computing for problem solving, December 28-30, 2012[C]. New Delhi: Springer, 2014.

[3] 于潇, 刘义, 柴跃廷, 等. 互联网药品可信交易环境中主体资质审核备案模式[J]. 清华大学学报(自然科学版), 2012, 52(11): 1518-1523.