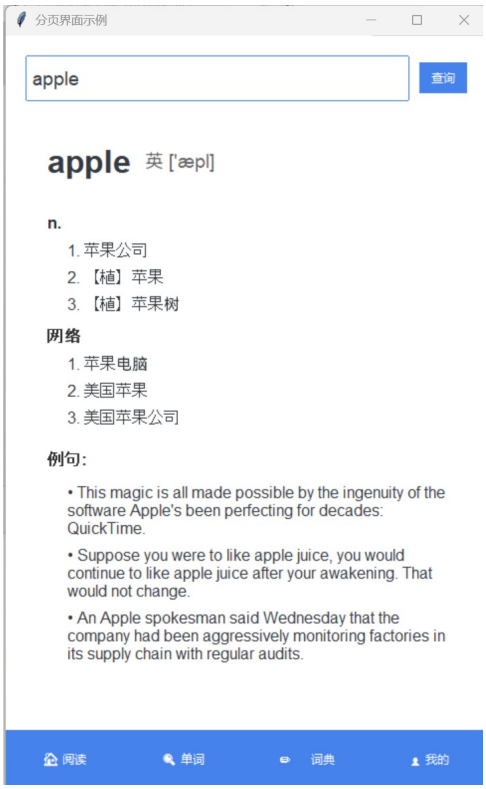
一、程序功能介绍

该程序用于帮助英语学习，包含文章搜索，单词学习，单词查询，显示个人资料管理等功能。

文章搜索:在搜索框内输入关键词后点击搜索按钮，改程序会从 https://www.enread.com 网站上爬取三篇相关文章，每篇文章下有收藏，点赞，阅读三个按钮，点击阅读按钮可以查看文章内容与翻译

单词学习：使用者重复学习十个单词，点击“显示释义”显示单词中文以及例句，下方四个按钮对应不同熟悉度，“不认识”对应 0%，“有点印象”对应 2.5%，“认识”对应 5%，“很熟悉”对应 7.5%，界面右上方显示总熟悉度和学习进度，以帮助使用者了解学习情况，点击四个按钮中的任意一个按钮以显示下一个单词。

单词查询：在搜索框中输入关键词然后点击查询或回车键，程序从 bing 网站上爬取单词具体信息。



个人资料管理：显示点赞收藏和发布的内容，点击“编辑资料”以修改用户名和 ID’

二、各模块与类设计细节

1. main.py

功能：项目的主程序入口，负责创建主窗口、配置样式和管理页面切换。

类设计：

App(tk.Tk)：继承自 tk.Tk，是应用程序的主窗口类。

__init__：初始化主窗口，设置标题、样式，创建容器框架、底部导航栏，初始化所有页面，并显示首页。

create_nav_buttons：创建底部导航栏的按钮，包括首页、发现、创作和我的按钮，点击按钮可切换页面。

show_frame：根据传入的页面名称，显示相应的页面。

2. ArticlePage.py

功能：提供文章推荐和搜索功能，支持文章点赞、收藏和阅读。

类设计：

ArticlePage(tk.Frame)：继承自 tk.Frame，是文章推荐页面类。

__init__：初始化页面，设置默认关键词，创建界面元素，并刷新文章内容。

create_widgets：创建顶部导航栏、搜索框和文章内容区域。

refresh_articles：清空现有文章内容，显示加载提示，使用线程获取文章数据。

_fetch_articles：在线程中调用 get_articles 函数获取文章数据，然后在主线程中更新 UI。

_update_article_display：更新文章显示，显示文章标题、摘要和操作按钮。

generate_summary：生成文章的 20 字左右摘要。

toggle_like：切换文章的点赞状态。

share_article：显示分享提示信息。

_show_error：显示获取文章失败的错误信息。

open_article：打开文章详情弹窗，显示文章内容。

3. WordPage.py

功能：提供单词学习功能，支持单词释义显示、熟悉度评价和进度更新。

类设计：

WordPage(ttk.Frame)：继承自 ttk.Frame，是单词学习页面类。

__init__：初始化页面，设置单词数据、用户学习进度，创建界面元素，并开始学习。

setup_ui：设置用户界面，包括顶部信息栏、单词显示区域和操作按钮区域。

next_word：显示下一个单词，隐藏释义和例句，更新进度。

toggle_meaning：切换显示 / 隐藏单词释义和例句。

rate_word：评价单词熟悉程度，更新单词记忆等级和下次复习时间。

update_progress：更新学习进度显示，包括总进度、今日学习数量和平均熟悉度。

4. DictionaryPage.py

功能：提供单词查询功能，从必应词典抓取单词释义、音标和例句。

类设计：

DictionaryPage(ttk.Frame)：继承自 ttk.Frame，是词典查询页面类。

__init__：初始化页面，创建界面元素。

setup_ui：设置用户界面，包括查询输入框、查询按钮和结果显示区域。

show_placeholder：显示初始提示信息。

show_loading：显示查询加载状态。

show_error：显示查询错误信息。

show_result：显示查询结果，包括单词、音标、词性释义和例句。

search_word：查询单词，显示加载状态，使用线程执行网络请求。

fetch_word_definition：从必应词典抓取单词释义，更新 UI 显示结果。

5. ProfilePage.py

功能：提供个人资料展示和编辑功能，支持用户信息的存储和管理。

类设计：

UserModel：用户数据模型类，负责数据存储与管理。

__init__：初始化数据文件路径，加载用户数据。

load_data：从文件加载用户数据，若文件不存在或加载失败，返回默认数据。

save_data：将用户数据保存到文件。

get_username：获取用户名。

get_userid：获取用户 ID。

get_following：获取关注数量。

get_followers：获取粉丝数量。

get_likes：获取获赞数量。

update_profile：更新用户资料并保存到文件。

EditProfileWindow(tk.Toplevel)：编辑资料弹窗类。

__init__：初始化弹窗，设置标题、大小，创建输入框和按钮。

center_window：将弹窗居中显示。

_save_changes：保存修改并关闭窗口，验证输入信息的有效性。

ProfilePage(ttk.Frame)：个人资料页面类。

__init__：初始化页面，创建用户数据模型，创建界面元素。

create_widgets：创建主框架、用户信息卡片、编辑资料按钮和内容标签页。

populate_example_content：填充示例内容到标签页。

open_edit_window：打开编辑资料窗口。

update_profile：更新用户资料并刷新界面，显示更新结果信息。

6. get_articles.py

功能：从指定网站搜索文章并提取内容。

函数设计：

get_search_results：根据关键词获取搜索结果页中的文章链接、标题和简介。

extract_article_content：根据文章链接提取文章内容，排除广告块。

get_articles：调用 get_search_results 和 extract_article_content 函数，获取指定数量的文章信息。

main：测试函数，接收用户输入的关键词，获取并输出文章信息。

7. user_data.json

功能：存储用户的个人资料信息，包括用户名、用户 ID、关注数量、粉丝数量和获赞数量。

三、小组成员分工

方山川：界面设计，单词学习模块，个人信息模块

张佳维：搜索文章模块

杨锦程：查询单词模块

四、项目总结与反思

总结

本次项目开发中，团队成员分工协作，成功实现了预期的功能目标。通过这次项目，我们积累了宝贵的开发经验，同时也认识到了存在的问题和不足。在未来的项目中，

我们将继续优化技术方案和协作流程，提高团队的开发效率和产品质量。

功能实现

成功实现了文章推荐、搜索、阅读、点赞和收藏功能，支持从网络获取真实文章数据。

开发了单词学习模块，实现了单词记忆追踪、进度统计和熟悉度评价功能。

实现了词典查询功能，可从必应词典获取单词释义、音标和例句。

设计并实现了个人资料管理功能，支持用户信息的编辑和保存。

技术亮点

采用 Tkinter 构建跨平台桌面应用，界面美观、交互流畅。

前后端分离的架构设计，提高了代码的可维护性和可扩展性。

后端使用 requests 和 BeautifulSoup 库实现网页内容爬取与解析。

实现了简单的单词记忆算法，基于艾宾浩斯遗忘曲线原理，提高学习效率。

技术难题

UI 与数据交互：前端与后端的数据传递和状态同步存在一定困难，特别是在处理异步操作时。通过定义清晰的 API 接口和使用回调机制解决了这一问题。

网页内容解析：不同网站的 HTML 结构差异较大，导致文章内容提取困难。通过编写自适应解析规则，提高了解析成功率。

性能优化：在处理大量文章数据和单词学习记录时，系统性能出现瓶颈。通过优化数据库查询和数据处理算法，提高了系统响应速度。

协作问题

沟通不畅：在开发过程中，偶尔会出现信息传递不及时或理解不一致的情况，导致部分功能需要返工。

接口定义模糊：在项目初期，API 接口定义不够清晰，导致前端与后端的集成工作花费了较多时间。后续通过编写详细的接口文档和进行接口评审，解决了这一问题。

反思与改进

技术方面

提前规划技术方案：在项目开始前，应更加充分地进行技术调研和方案设计，避免在开发过程中频繁调整技术路线。

加强代码规范：制定统一的代码规范和注释标准，提高代码的可读性和可维护性。定期进行代码审查，及时发现和纠正问题。

引入自动化测试：增加单元测试和集成测试，确保代码质量和功能稳定性。自动化测试可以提前发现问题，减少后期调试的工作量。

协作方面

强化沟通机制：建立更加高效的沟通机制，确保团队成员之间信息畅通。定期组织技术分享和团队建设活动，增强团队凝聚力。

优化分工协作：根据团队成员的技能优势和兴趣爱好，更加合理地分配任务。明确各成员的职责和工作边界，避免任务重叠或遗漏。

加强项目管理：使用专业的项目管理工具，如 Jira、Trello 等，对项目进度、任务和问题进行跟踪和管理。制定详细的项目计划和里程碑，确保项目按计划推进。