SysY-Formatter

摘要

本项目是一个针对 SysY 编程语言的代码分析与处理工具，核心功能为代码高亮显示和自动化格式化。通过实现词法分析与语法解析基础模块，可将无格式的 SysY 源代码转换为高可读性的 HTML 高亮版本，并按照合适的代码规范自动调整代码结构。本项目适合练习词法分析、语法处理的综合小项目，2 小时可完成基础任务。

本项目的输入应当为以 `.sy` 为文件名后缀的 SysY 源代码，输出为 HTML 文档。如果你认为有必要，可以搭配 CSS 样式表和 JavaScript 脚本。

项目 `code` 目录下提供了若干 SysY 源代码作为样例输入。你也可以自己构造合适的用例输入。

你可以使用 C++、Java、Python 或其他任何方式实现本项目的功能。机房预安装了一些编程环境，如果你需要安装任何依赖包或库，你可以直接访问互联网安装。

关于本项目的任何术语、实现方式，都可以上网检索或询问大模型。如果没有得到想要的答案，可以询问工作人员。

1. 基础任务一：SysY 语言词法识别

SysY 语言是本项目使用的编程语言，是 C 语言的一个子集。换言之，SysY 定义的所有文法规则（包括词法规则和语法规则）都是 C 语言允许的。如果你已经熟悉 C 语言，你完全可以把 SysY 语言当成 C 语言的一部分来使用。

附录一：SysY语言定义.pdf 中“SysY 语言的终结符特征”这一节给出了 SysY 语言词法的形式化定义。

类别	描述	示例
关键字	语言保留关键字	`const`, `int`, `void`, `if`, `else`, `while`, `break`, `continue`, `return`
运算符	算术、关系、逻辑、赋值	`+`, `-`, `*`, `/`, `%`, `==`, `!=`, `<`, `>`, `<=`, `>=`, `&&`, `||`, `!`, `=`
分隔符	界定语句与表达式边界	`;`, `,`, `(`, `)`, `{`, `}`, `[`, `]`
标识符	用户定义的名称，满足 C 语言标识符规范 正则表达式为 `[_a-zA-Z][_a-zA-Z0-9]*`	`main`, `x`, `i`, `_temp`
整型常量	十进制、八进制、十六进制整数	`0`, `123`, `077`, `0x3F`
注释	单行和多行注释	`// comment`, `/* block */`
空白符	空格、制表符、换行符等	`\t`, `\n`, `\r`, 空格

上表给出了一些常见的词法示例。

2. 基础任务二：代码高亮

将代码输出到 html 中，将不同的词法单元用合适的标签框定，并优化可视化效果。

项目目录下的 output/test2.html 是一个经过高亮处理的代码样例，但还不够美观。请你设计合适的高亮展示方案。以下是一些可能有帮助的建议：

- 一般来说，代码使用等宽字体显示，但在注释中可以使用适合一般文本阅读的字体；请注意区分衬线字体和无衬线字体的区别；
- 不同的词法单元应该用不同的颜色显示，但同一方案中不应出现高对比配色（例如红-绿、蓝-黄）；
- 不同层级的括号可以使用不同的方案，以便用户识别配对的括号；
- 你可以访问大模型以获取其他的代码高亮方案建议。

3. 基础任务三：代码格式化

按照你喜欢的风格进行格式化代码，使得代码美观易读。

项目根目录下的 附录二：华为C语言编程指南.pdf 提供了一种工业界的代码风格规范，以供参考。

你可以访问大模型以获取其他的代码格式化方案建议。

4. 进阶任务四：SysY 语法规则检测

附录一：SysY语言定义.pdf 中给出了关于 SysY 语言完整的语法定义。

该任务包括：

1. 语法树构建：基于 SysY 文法定义，构建语法分析器；
2. 错误提示：当输入源代码不满足语法规则时，输出清晰的错误消息，提示行号、列号、预期文法结构；
3. 错误恢复：尝试最小化中止解析行为，在可能的情况下跳过错误部分继续分析；
4. 支持错误分级：
    - 严重错误：如语法树无法构建；
    - 一般错误：如缺少分号；
    - 建议：如空函数体建议添加注释。

5. 进阶任务五：标识符定义寻找

本功能旨在模拟 IDE 中的“跳转到定义”功能，具体需求如下：

1. 符号表构建：
    - 在语法分析过程中建立作用域和标识符符号表；
    - 每个标识符应记录其定义位置（文件名、行号、列号）；
2. 上下文识别：
    - 能够区分变量名、函数名、数组名；
    - 支持局部变量、全局变量的作用域区分；
3. 跳转定位：
    - 用户点击源代码中的标识符后，前端发送请求；
    - 后端响应该标识符的定义位置；
    - 如果是重名标识符，应优先匹配最近作用域；
4. 前端支持（可选）：
    - 利用 `<span data-ident="xxx">` 的结构封装标识符；
    - 鼠标悬停高亮显示定义信息（如 `int x defined at line 10`）；
    - 点击后跳转到定义位置，可通过锚点或脚本实现滚动跳转。