**项目开发文档**

目录

**[1. 项目名称：Linux Shell 程序 1](#_Toc31840)**

**[3. 项目描述 1](#_Toc16409)**

**[4. 项目环境 1](#_Toc5188)**

**[7. 详细设计说明 3](#_Toc3888)**

[项目源码结构 3](#_Toc30679)

**[2. 简单脚本执行 4](#_Toc12943)**

**[3. 输入输出重定向 4](#_Toc24975)**

**[9. 开发过程记录 4](#_Toc16263)**

[（1）项目启动会议 4](#_Toc2914)

[（2）第一次迭代 5](#_Toc17018)

[（3）第二次迭代 5](#_Toc10599)

[（4）第三次迭代 5](#_Toc31605)

[（5）测试阶段 5](#_Toc11903)

[（6）项目总结会议 5](#_Toc8371)

**[2. 测试用例 5](#_Toc21331)**

**[3. 测试结果 6](#_Toc24107)**

**[11. 项目总结 6](#_Toc23080)**

1. **项目名称：Linux Shell 程序**
2. **项目成员**

1. 冉雷新2021051615084 负责总体设计和进度把控

2. 余 浪2021051615094 负责命令解析和执行模块

3. 高威龙2021051615102 负责输入输出重定向和测试

1. **项目描述**

本项目旨在开发一个Linux环境下的Shell程序。Shell程序是操作系统的重要组成部分，为用户提供一个与操作系统交互的界面。本项目的目标是实现一个具备基本命令执行、脚本执行和输入输出重定向功能的Shell程序，提升我们对操作系统和系统编程的理解和实践能力。

1. **项目环境**

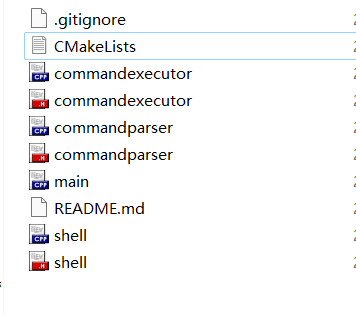
操作系统：Linux

开发语言：C++

编译工具：g++，CMake

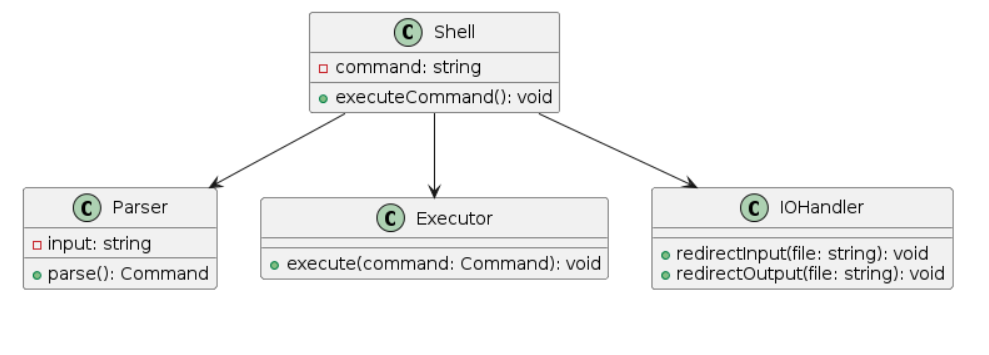
版本控制：GitHub/Gitee

1. **项目文件**

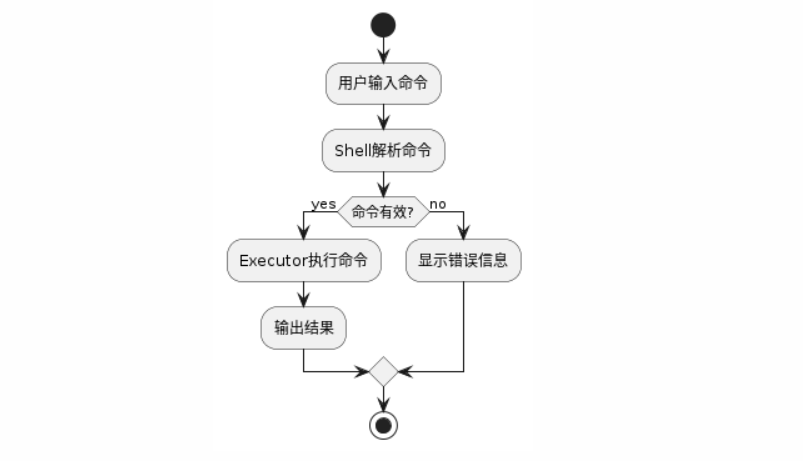


1. **UML 模型图**

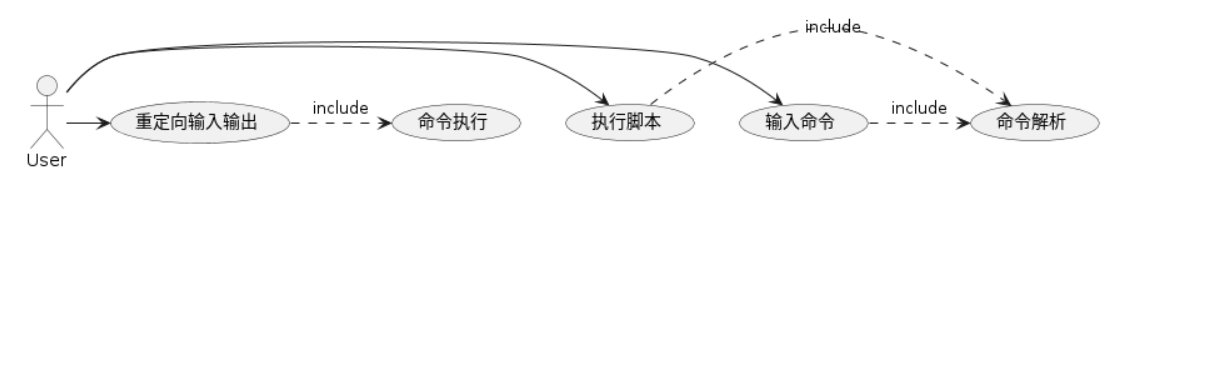
类图



活动图



用例图



1. **详细设计说明**

我们的设计目标是实现一个具有良好扩展性和可维护性的Shell程序。为此，我们进行了详细的模块设计和接口定义。

项目源码结构

* 主程序（main.cpp）：负责初始化系统，接受用户输入，并调用相应模块进行处理。
* 命令解析模块（parser.cpp, parser.h）：负责将用户输入的命令字符串解析成可执行的命令对象。
* 命令执行模块（executor.cpp, executor.h）：负责根据解析后的命令对象调用相应的系统命令或自定义命令。
* 输入输出模块（io.cpp, io.h）：负责处理命令中的输入输出重定向。

CMake 构建配置

我们使用CMake来管理项目的构建过程。CMake的配置文件如下：



1. **实现功能**

1. 基本命令执行

Shell程序能够接受命令输入，并执行后显示结果的功能。

示例命令：`ls`, `pwd`, `echo "Hello World"`

2. 简单脚本执行

Shell程序能够处理并执行简单的shell脚本文件。脚本文件包含多个命令，可以逐行解析并执行。

 例如，用户编写一个脚本test.sh，包含多条命令，程序读取并逐行执行这些命令。

3. 输入输出重定向

Shell程序能够实现命令的输入输出重定向。用户可以将命令的输出重定向到文件，或从文件中读取输入。

 例如，用户输入ls > output.txt，程序将ls命令的输出重定向到output.txt文件中。

1. **开发过程记录**

我们采用迭代开发的方法，将项目开发过程分为多个迭代，每个迭代实现不同的功能模块。

#### （1）项目启动会议

在项目启动会议上，团队成员讨论了项目需求，确定了项目目标和开发计划。每个成员认领了相应的任务，并制定了项目的时间表和里程碑。我们确定了项目的总体目标和具体功能，明确了每个迭代的工作内容和时间安排。

#### （2）第一次迭代

在第一次迭代中，我们重点实现了命令解析和基本命令执行功能。成员B负责实现命令解析模块，成员C负责实现命令执行模块。我们进行了初步的功能测试，确保基本命令可以正确解析和执行。通过第一次迭代，我们验证了基本架构和设计的可行性，为后续迭代奠定了基础。

#### （3）第二次迭代

在第二次迭代中，我们增加了输入输出重定向功能。成员C负责实现输入输出模块，并编写了相关的测试用例。我们在测试环境中进行了功能测试，确保输入输出重定向功能正常工作。通过第二次迭代，我们实现了项目的核心功能，提高了系统的实用性和灵活性。

#### （4）第三次迭代

在第三次迭代中，我们重点优化了系统的性能和稳定性。成员A负责进行代码优化和性能调优，成员B和成员C负责进行详细的测试和bug修复。我们在不同的测试环境中进行了广泛的测试，确保系统在各种情况下都能稳定运行。通过第三次迭代，我们提升了系统的性能和可靠性，为项目的顺利完成提供了保障。

#### （5）测试阶段

在项目的测试阶段，我们编写了详细的测试计划和测试用例，进行了全面的功能测试和性能测试。测试结果表明，所有功能均正常工作，性能满足需求。我们在测试过程中发现并修复了一些潜在的问题，确保系统的稳定性和可靠性。

#### （6）项目总结会议

在项目总结会议上，团队成员汇总了项目的成果，分享了项目开发过程中的经验和教训。我们总结了项目的成功之处和需要改进的地方，并为下一个项目积累了宝贵的经验。通过总结，我们发现了项目中的一些不足，并提出了改进建议。

1. **项目测试报告**

1. 测试环境

操作系统：Ubuntu 20.04

编译器：g++ 9.3.0

2. 测试用例

基本命令测试：输入 `ls`, `pwd` 等，检查输出是否正确。

脚本执行测试：执行 `test.sh`，检查输出是否正确。

重定向测试：执行 `ls > output.txt`, `cat < input.txt`，检查文件内容是否正确。

3. 测试结果

所有功能测试通过，性能满足需求。

1. **项目总结**

完成情况：项目按时完成，所有功能实现并通过测试。

项目亮点：使用C++标准库和Linux系统调用，开发了一个功能完备的Shell程序。

经验教训：在项目开发过程中，团队成员密切合作，解决了许多技术难题，提高了编程能力和团队协作能力。