# 项目简要开发文档

### 1 Shell项目简介

本项目旨在开发一个简单的 Linux Shell 程序，支持基本的命令执行、管道操作、输入/输出重定向等功能。使用 C++ 标准库进行高级数据结构和算法操作，同时使用 Linux 系统调用来实现底层的进程管理和文件操作。

### 2 小组分工

聂安芹：Shell基本命令和脚本执行，文档编写；

向禹涵：实现IO重定向，PPT制作，git仓库创建；

代鑫 ：实现管道功能，if-else流程控制功能，答辩视频录制。

### 3 功能要求

命令执行: 能够执行用户输入的基本命令，如 ls、pwd等。

管道支持: 能够处理简单的管道操作，如 command1 | command2。

输入输出重定向: 支持 <、>、>> 等符号实现输入和输出的重定向。

后台执行: 支持将命令放入后台执行，如 command &。

信号处理: 能够处理常见的 Shell 信号，如 Ctrl+C 终止进程等。

### 4 技术选择

编程语言: C++ 作为主要编程语言，利用其面向对象的特性和标准库的丰富功能。

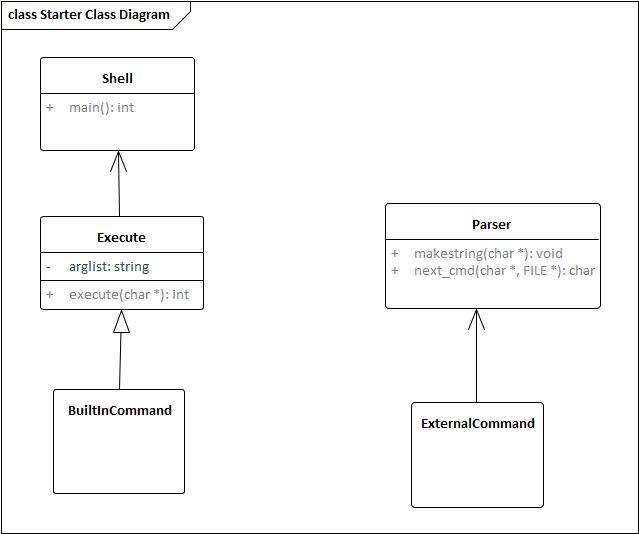
系统调用: 使用 Linux 提供的系统调用来管理进程、文件、管道等系统资源。

数据结构: 使用malloc函数和 STL 提供的数据结构，如 td::string 等来处理命令解析、参数传递等。

命令解析: 使用字符串操作和正则表达式来解析用户输入的命令和参数。

### 5 设计概要

##### 5.1 uml类图



**5.2 概要设计**

命令解析:

使用 malloc函数和 std::string 来解析用户输入的命令和参数。

支持空格分隔、引号包围的参数解析等。

执行命令:

使用 fork()、execvp() 等系统调用创建和执行新进程。

处理前台命令和后台命令的区别。

管道操作:

使用 pipe() 系统调用来创建管道。

将一个进程的输出连接到另一个进程的输入。

输入输出重定向:

使用 dup2() 系统调用来重定向文件描述符。

环境变量扩展:

使用 getenv() 系统调用获取环境变量。

在命令执行前替换命令中的环境变量。

信号处理:

使用 signal() 系统调用来处理信号，如 Ctrl+C。

忽略或捕获特定信号，确保 Shell 的稳定性和可靠性。

### 6 实现步骤

初始化:

设置 Shell 环境，如加载环境变量。

显示提示符等待用户输入。

解析用户输入:

使用字符串操作和正则表达式解析用户输入的命令和参数。

检测管道、重定向符号等特殊操作。

执行命令:

根据解析得到的命令类型，选择合适的执行方式。

使用 fork() 创建新进程并使用 execvp() 执行命令。

处理管道和重定向:

如果命令包含管道或者重定向，设置合适的管道、重定向。

使用 dup2() 系统调用重定向标准输入和输出。

后台执行:

如果命令以&结尾，将命令放入后台执行。

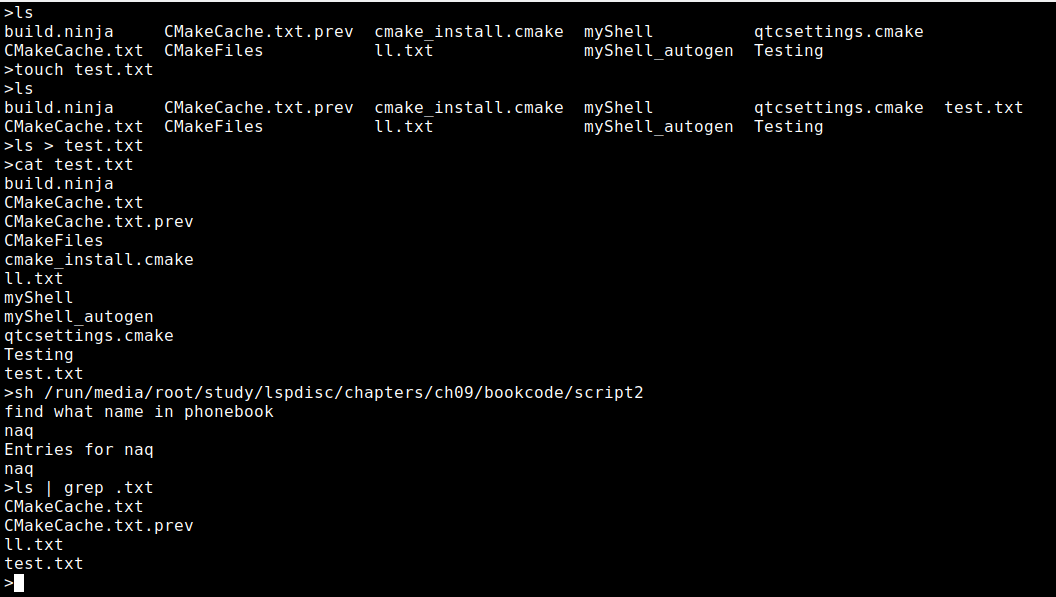
信号处理:

处理 Ctrl+C 等信号，确保 Shell 能够优雅地终止或忽略信号。

清理和退出:

在 Shell 收到终止信号时，清理资源并退出。

**7 运行结果**



### 7 结论

开发一个 Shell 程序涉及到系统调用、进程管理、输入输出操作等多方面的知识，结合 C++ 标准库提供的高级数据结构和算法，可以使 Shell 更加稳定和高效。