Project 1: A Simple Shell

- 本次课程项目的负责助教为蔡子诺 zinuocai@gmail.com。
- 在完成课程项目的过程中有任何问题，欢迎在Canvas对应的讨论区进行提问。

1. 基本功能

Shell是一个用C语言编写的程序，它是用户使用Linux的桥梁。Shell既是一种命令语言，又是一种程序设计语言。通俗易懂的Shell教程有：

- Shell教程 | 菜鸟教程
- Bash 脚本教程

我们的的第一个项目是编写一个简单的Shell工具**myshell**，它具有以下属性：

1. 它可以支持以下的内部指令：
    - `cd <directory>`
        - 从当前的默认目录跳转到指定的目录`directory`。
        - 如果没有指定参数`directory`，输出当前目录。
        - 如果`directory`目录不存在，输出错误信息。
        - 该指令同时要修改环境变量`PWD`。
    - `clr`：清空屏幕上的显示内容。
    - `dir <directory>`：输出目录`directory`下的所有内容，包括目录和文件。
    - `environ`：输出所有的环境变量。
    - `echo <comment>`
        - 输出`comment`的具体内容并换行。
        - 当`comment`的内容出现多个space或者tab时，应该压缩为一个space。
    - `help`：输出用户指南。
    - `pause`：停止Shell的执行，直到用户按下`Enter`键。
    - `quit`：退出。
    - Shell的环境变量中应该包含`shell=<pathname>/myshell`。其中`<pathname>/myshell`是Shell工具的绝对路径。
2. 所有的其他指令可以通过调用`fork`与`exec`完成。父进程通过`fork`生成子进程，子进程调用`exec`方法执行用户的指令。注意，子进程在执行时应该包含环境变量`parent=<pathname>/myshell`，`<pathname>/myshell`的定义同上。
3. 如果Shell工具在使用时带有参数，那么它可以从参数指定的文件中读取指令，并依次执行。例如，当我们这样使用Shell工具时：

    ```
    myshell batchfile
    ```

    那么，我们会依次读取`batchfile`文件的每一行并执行。当读取到文件的最后一行时，Shell会退出。
4. 它可以支持输入输出重定向。例如，当我们在Shell中执行以下指令：

    ```
    programname arg1 arg2 < inputfile > outputfile
    ```

或者以下指令：

```
programname arg1 arg2 < inputfile >> outputfile
```

- 其中，`programname`是可执行指令，`arg*`是指令的参数。该指令从`inputfile`中获取用户输入，而不是标准输入`stdin`；指令执行的结果会输出到`outputfile`，而不是标准输出`stdout`。
- 输出重定向会和`dir` `environ` `echo` `help`含有输出的内部指令共同使用，输出结果会重定向到用户指定的文件中。例如`help > outputfile`的执行结果应该输出到用户指定的文件`outputfile`。
- 当使用输出重定向时
  - 如果表示重定向的字符串是`>`，
    - 如果`outputfile`不存在，则新建文件。
    - 如果`outputfile`存在，那么输出会覆盖原文件。
  - 如果表示重定向的字符串是`>>`，
    - 如果`outputfile`不存在，则新建文件。
    - 如果`outputfile`存在，那么输出添加到原文件后面。

5. Shell可以支持后台任务的执行。当一条指令后面有`&`符号时，Shell不需要等待该指令执行结束才能返回。
6. Shell工具的`prompt`应该包含当前目录的地址。

注意：我们可以假设所有命令行参数（包括重定向符号`<`，`>`和`>>`和后台执行符号`&`）通过空格（一个或多个space和tab）与其他命令行参数隔开。

2. 项目进度

该项目的完成方式为讲练结合：教师负责对应课程内容的讲解，助教负责项目的技术指导。为了帮助同学们完成该项目，助教将组织三次习题课，逐步完善整个项目。三次习题课的内容包括：

进度	内容
一	Shell简介与项目介绍。 完成`clr` `dir` `environ` 与 `quit` 功能。
二	环境变量与 `cd` 功能。 `fork` 与 `exec` 功能。
三	输入输出重定向。 功能完善与项目评测。

3. 项目要求

1. 设计一个满足上述条件的简单命令行Shell。
2. 写一本简单的手册描述如何使用你的Shell。该手册应包含足够的细节，供UNIX初学者使用。例如，它解释输入输出重定向、程序环境变量和后台程序执行的概念。手册必须命名为`readme`，并且必能够被标准文本编辑器阅读。

   `readme`中不需要添加代码等无关文件，可以参考`csh`与`zsh`的用户手册，当然我们的Shell没有它们那么复杂。
3. 源代码必须有便于理解的注释与代码结构，便于助教评审。

4. 提交文件应仅包含源文件`myshell.c`、`makefile`和`readme`，不需要包含可执行程序。请将上述三个文件压缩为zip格式，命名为`[学号].zip`并上传到Canvas的作业区。

4. 评分标准（满分150分）

- 通过`makefile`对项目进行编译，没有警告或者报错（5）
- 支持从键盘输入指令和从文件读取指令（10）
- 内部指令正常执行（30）
- 外部指令正常执行（10）
- 输入输出重定向（10）
- 后台执行（10）
- 代码结构规范、注释完整与可读性（25）
- 用户手册（50）
    - 内部指令的使用说明（10）
    - 环境变量的说明（10）
    - 输入输出重定向的说明（10）
    - 后台执行的说明（10）
    - 结构规范，便于理解（10）