程设大作业：文件系统使用手册  
分工：all by李晨昊

设计思路：

所有文件相关操作由FileManager类管理。其构造函数接受一个参数表示文件系统所在位置，如果文件不存在则创建文件并初始化系统。其析构函数会向文件系统中完整写入内存中的相关内容（每次写操作后都会写入变化，所以这其实是不必要的）。

FileManager仅有一个public成员函数exec，它接受一个std::vector<std::string>作为参数。第0个字符串表示指令名称，此后的字符串表示具体参数。exec返回一个std::string来表征操作的状态（成功/失败）并在主函数中输出。

Util文件中定义了一些与FileManager本身状态无关的功能函数

Zip文件中定义了Zipper类和DeZipper类，两个类都重载了operator(std::string)，用于实现文件的压缩和压缩文件的展开。假定zip和dezip分别为两个类的对象，则始终保证dezip(zip(str))==str

使用方法

统一说明：

1. 要求path存在而path不存在的，输出path+“ No such file or directory“
2. 要求path为文件而path存在且不为文件的，输出path+“ is not

a file.”要求path为文件夹的同理。

1. 涉及到创建文件夹而文件夹已经存在的输出path+“ already exists.”。
2. 文件夹名/文件长度超出题目所给出的限制时输出path/content+“ is too long”。文件夹名保存了它的绝对路径，长度为绝对路径长度。
3. 考虑到读者智力水平足够高，部分简单操作不给出示例。
4. 基本操作

>>exit

退出文件系统，保存一切修改。

注意：直接关闭控制台应用会导致未保存的修改丢失

可能会导致其他未定义后果，本程序不做保证

>>cd $path

将当前文件夹改至path对应的文件夹。

1. 读操作

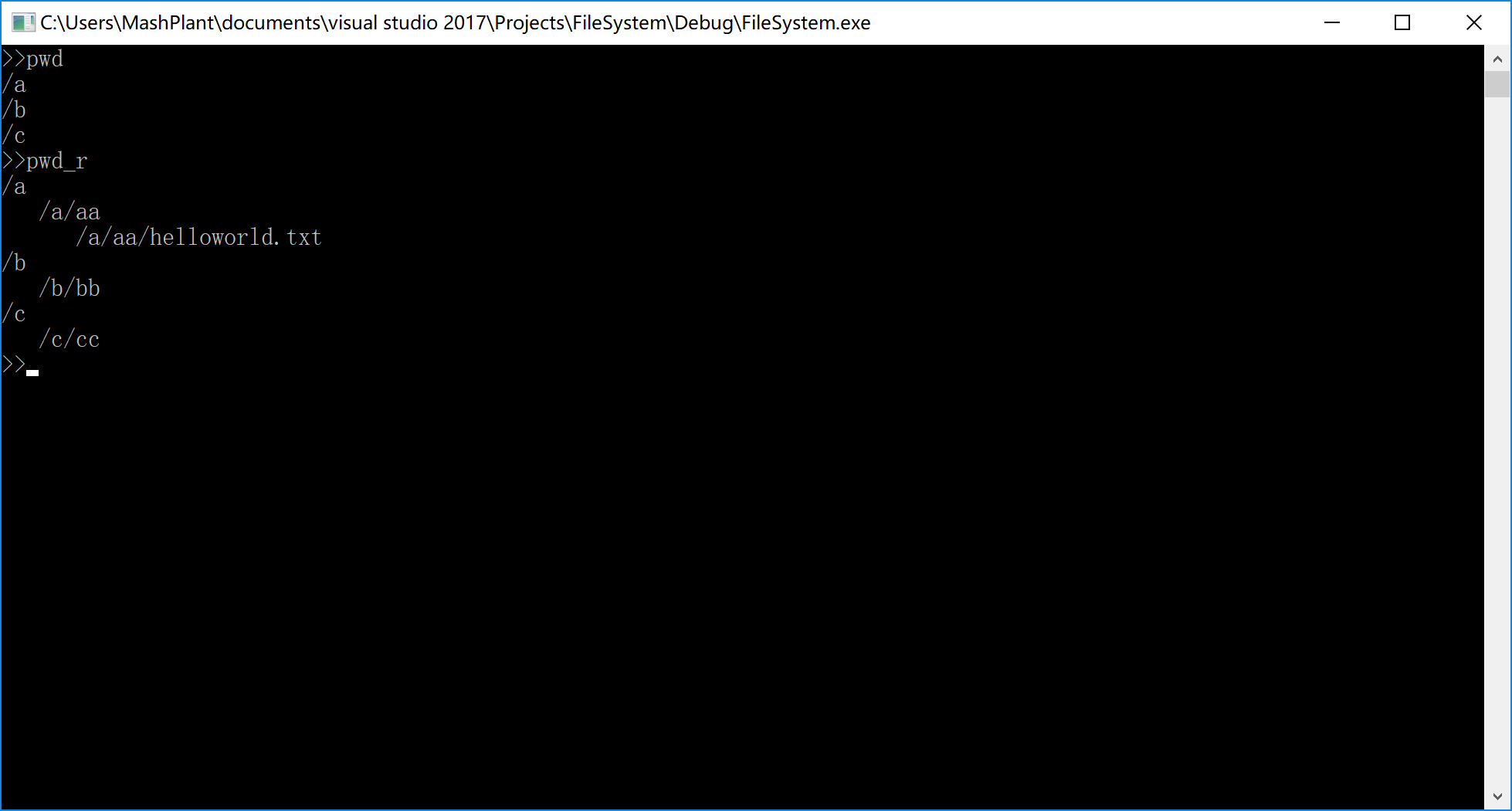
>>pwd

显示当前文件夹下的文件夹和文件

>>pwd\_r

递归地显示当前文件夹下的所有文件夹和文件

示例（包含pwd）：



>>ls $path

查看path对应的文件或文件夹。

第一行输出path的类型（文件/文件夹）。

第二行输出path的大小。

大小的定义为：

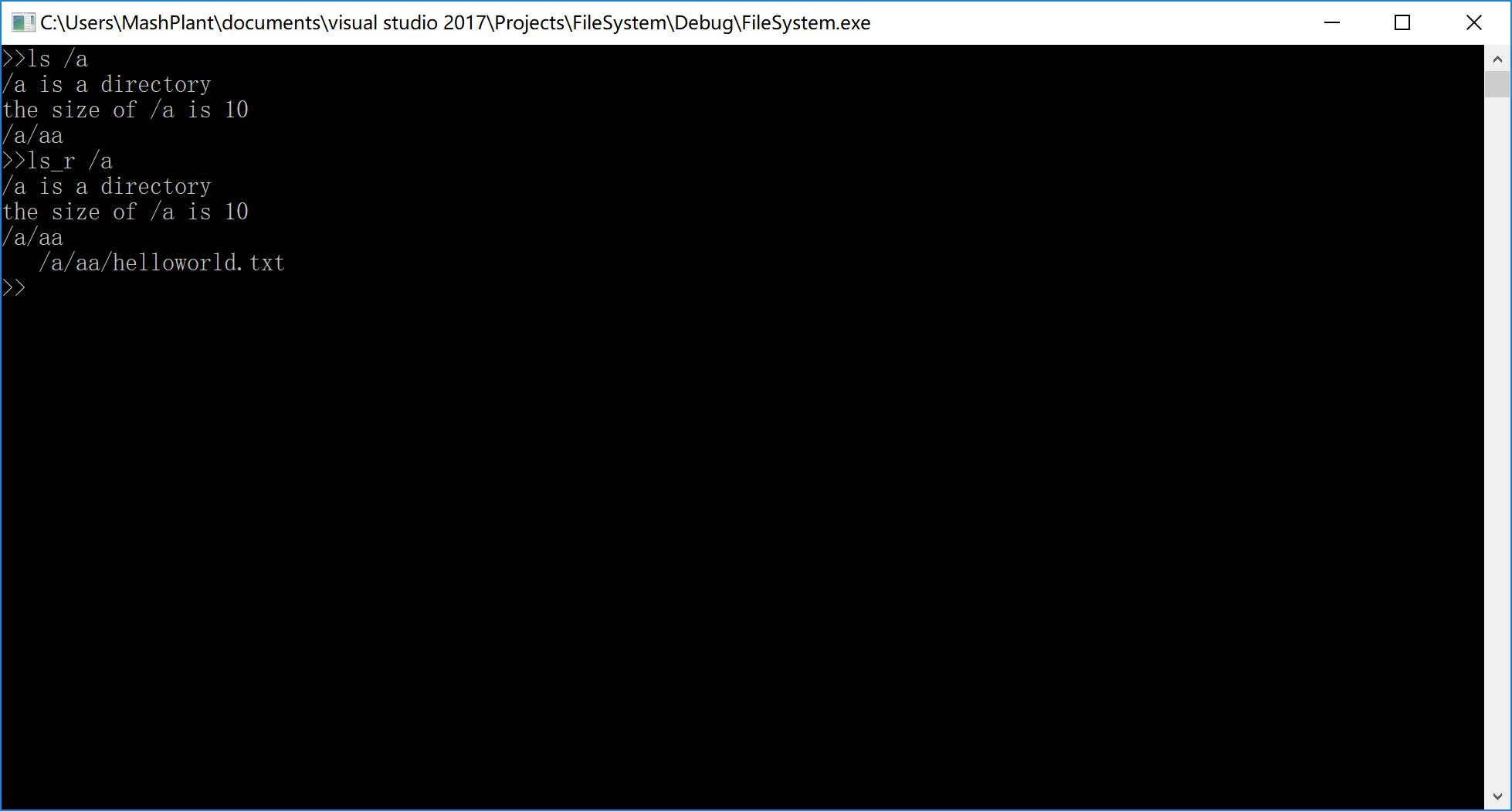
size(path) = path.data.size if isinstance(path,File) else sum([size(i) for i in path.child])

此后若干行，若path为文件夹，输出path下的文件和文件夹；若path为文件，输出文件内容。

>>ls\_r $path

类似ls $path，区别是若path为文件夹时，将递归地显示path下的所有文件夹和文件

示例（包含ls）：



>>cat $path

输出path对应的文件的内容。

1. 写操作

说明：所有的写操作在完成后都会提示“write the change?y/n”，输入y表示将刚才的改动写入文件。当然，即使这里不写入，改动也会在此后某次写入文件中被写入。

>>mkdir $path

创建path目录。

>>echo $str $path

将$str 写入$path 文件。如果文件已存在则覆盖原文件，否则创建文件

>>rm $path

删除$pat文件

>>rmdir

递归地删除$path目录下的所有文件和文件夹

注意：若删除当前目录，则当前目录自动跳转到根目录。

>>cpy $path1 $path2

将文件path2的内容复制到文件path1

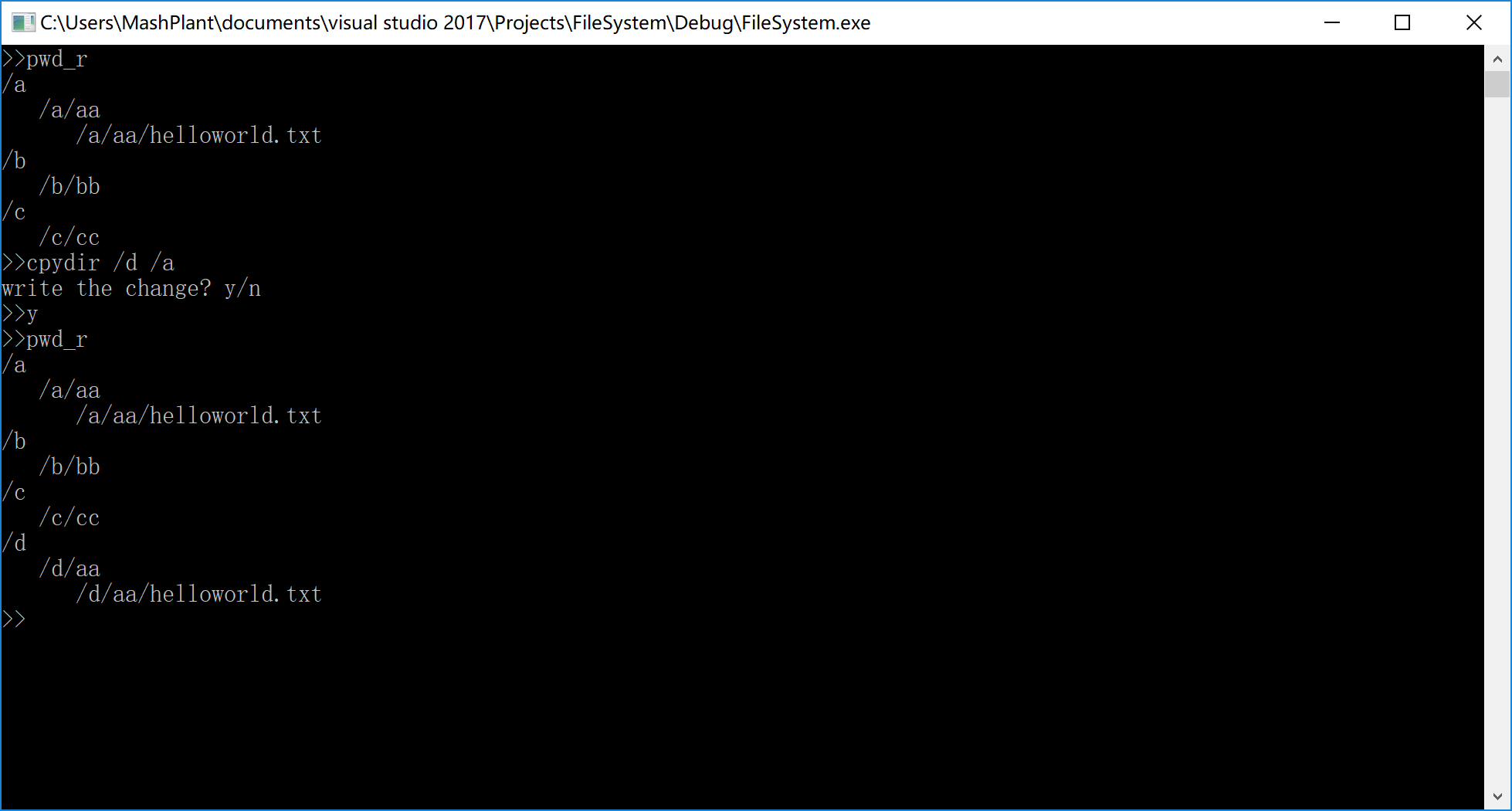
注意：path1和path2不能指向同一个文件，否则将做出提示。

>>cpydir $path1 $path2

创建文件夹path1，递归地将path2下的的文件和文件夹复制到path1下

注意：path1不能是path2的子目录，否则将做出提示。

示例：



1. 查找操作

>>find $str

从根目录开始查找所有名字包含str的文件和文件夹并显示。

>>find\_re $str

从根目录开始查找所有名字中 含有能被正则表达式str匹配的字串 的文件和文件夹。

示例：

