思路

阶段一 identifyinput,readfile

1，通过argc，argv识别输入的指令

2，写函数来识别指令中的内容，包括—help，-v、--verbose，[target]；

3，fopen：类型为“r”，fgets，用while逐行读，逐行修整

4，fopen：类型为“w”，识别到-v…时输出

5，在3的基础上添加检查语法语句，第一次出现的不能是命令，通过\t识别命令，其余都是target,然后检查冒号

阶段二 analysis

1. 构造结构体rule储存解析逐行，并检查依赖是否存在和定义；结构体的内容：目标名，依赖名s，依赖数量，命令s，命令数量。
2. 先用两个大字符串数组把目标行和命令行分别对应存起来，一个目标对应多个命令 存到同一行里，后续可以通过\t拆开
3. 写一个函数把存上的目标行的目标和依赖拆开存到结构体，然后把对应的命令行也拆开存上，用一个循环全部读进去
4. 再写一个检查函数，把存好的依赖分别拿出来和现有文件和目标们比对
5. 通过第三个不包含‘-’的参数指定要构建的目标或对应指令，执行指令之后提前结束程序

阶段三 order

1. 图，节点，边
2. 先构建依赖图，然后排序，然后按顺序检查构建时间并按需要构建目标，然后释放依赖图的内存
3. 在主程序中使用两层循环添加边（依赖→目标）到图，在添加边时添加点到图
4. 添加点的函数要先检索一遍是否已有节点，有则返回一个索引，没有则创建节点，接入表中，初始化入度
5. 在添加边的函数中实现添加边到邻接表，然后更新对应节点的入度
6. 排序：先将入度为0的目标或依赖加入队列，输出这些，然后找到他们指向的节点的入度，然后再削减入度，然后再把入度为0的加入队列，直到队列的前后指针碰一起，结束
7. 构建时，先将入度为0的目标或依赖加入队列，检查其依赖和是否需要再重新构建，需要则执行构建，输出这些，然后找到他们指向的节点的入度，然后再削减入度，然后再把入度为0的加入队列，直到队列的前后指针碰一起

阶段四 process

1. 用新函数代替System的功能
2. 创建子进程，在子进程中使用execvp，在父进程中等待子进程，检查子进程的状态
3. 注意execvp的参数

阶段五 variable

0，建立一个哈希表，在读入文件时读入并储存变量

1. 添加一个函数，在解析时对每一句内容进行识别
2. 添加一个函数，在识别成功之后对这一行的内容进行替换
3. 在上一个函数中添加识别函数和其自身进行检测和递归替换
4. 如果哈希表中没找到要替换的变量，报告此变量未定义，然后输出错误返回值以结束程序
5. 可以写一个小函数来在哈希表中检索，以增加可读性