makefile记录

1. makefile并不是脚本，而是告诉make命令依照什么样的规则去编译和连接程序。makefile由三部分组成，目标（target），目标的依赖（prerequisites），以及生成目标的所要执行的操作（command）。因此在makefile中，有意义的会执行的shell命令也就存在于command部分，直接放在最开始时没有意义的。但是在最前面可以定义makefile文件本身的变量。
2. make在执行command前会把执行的命令输出，称之为回显。如果要执行的命令以@开始，则不会回显：

   回显：

   ```
   target:
     echo hello,world!
   ```

   make输出：

   ```
   echo hello,world!
   hello,world!
   ```

   不回显：

   ```
   target:
     @echo hello,world!
   ```

   make输出：

   ```
   hello,world!
   ```

3. make额外提供了一些函数，函数调用后，函数的返回值可以当作变量来使用。函数的调用语法和变量是类似的，要提前告诉make这个名字不是一个普通的字符串：

   ```
   $(<function> <arguments>)
   ```

   这里`<function>`是函数名，`<arguments>`为该函数的参数，参数间用逗号分隔，函数名和参数之间用空格分隔。因为这个命令的返回值是函数运行的结果，所以实际上就可以把整体当作一个变量，比如这个为字符串增加后缀的`addsuffix`函数：

```
files = $(addsuffix .c, foo bar)
target:
  @echo ${files}
```

输出为：

```
foo.c bar.c
```

makefile里还提供了一个长得很容易混淆的`shell`函数，他的参数是执行操作系统shell的命令，返回这个命令执行后的标准输出。因为这个不是执行shell语句本身，所以把它放到最开始也是可以运行的：

```
hello = $(shell echo hello,world!)
target:
  @echo ${hello}
```

输出为：

```
hello,world!
```