文件处理

Ref Pseudocode

Writing to a file

一个文件在项目结束后保留数据。0478 使用四个命令，且每个命令都命名了其操作的文件。

```
OPENFILE <file identifier> FOR WRITE
WRITEFILE <file identifier>, <data>
CLOSEFILE <file identifier>
```

```
OPENFILE "names.txt" FOR WRITE
WRITEFILE "names.txt", "Mei"
WRITEFILE "names.txt", "Sam"
CLOSEFILE "names.txt"
OUTPUT "Saved"
```

三种**文件模式**文件模式：

模式	功能说明
WRITE	创建一个新文件 – **原有内容全部丢失**
APPEND 追加	在现有内容末尾添加
READ	从头开始读取

```
OPENFILE "log.txt" FOR WRITE
WRITEFILE "log.txt", "first"
CLOSEFILE "log.txt"
OPENFILE "log.txt" FOR APPEND
WRITEFILE "log.txt", "second"
CLOSEFILE "log.txt"
OUTPUT "Both lines saved"
```

- 一个 WRITEFILE 写入**一行**。
- CLOSEFILE 是必须的。未关闭的文件可能导致尚未写入的数据丢失，这是常见的考试考点。写入，这属于标准考题答案。

常见错误

- 将 FOR WRITE 误作 FOR APPEND 使用，后者会静默覆盖原有内容。
- 忘记 CLOSEFILE。
- 打开一个已经打开的文件。

Reading a file to the end

READFILE 将一行读入变量中。该变量必须先声明。

```
OPENFILE <file identifier> FOR READ
READFILE <file identifier>, <variable>
CLOSEFILE <file identifier>
```

通常你无法预知文件有多少行，因此需逐行读取**直到文件末尾**文件**结束** — 这正是 EOF 函数所报告的状态。

```
OPENFILE "names.txt" FOR WRITE
WRITEFILE "names.txt", "Mei"
WRITEFILE "names.txt", "Sam"
WRITEFILE "names.txt", "Ana"
CLOSEFILE "names.txt"

DECLARE Line : STRING
OPENFILE "names.txt" FOR READ
WHILE NOT EOF("names.txt") DO
    READFILE "names.txt", Line
    OUTPUT Line
ENDWHILE
CLOSEFILE "names.txt"
```

EOF("names.txt") 在每行读取完毕后变为 TRUE，因此 WHILE NOT EOF(...) 意味着 当仍有内容可读时。



打开文件用于写入或读取，逐行使用，然后关闭

■ 边读边计数

```
OPENFILE "marks.txt" FOR WRITE
WRITEFILE "marks.txt", 40
WRITEFILE "marks.txt", 65
WRITEFILE "marks.txt", 88
CLOSEFILE "marks.txt"

DECLARE Mark, Count, Total : INTEGER
Count ← 0
Total ← 0
OPENFILE "marks.txt" FOR READ
WHILE NOT EOF("marks.txt") DO
    READFILE "marks.txt", Mark
    Count ← Count + 1
    Total ← Total + Mark
ENDWHILE
CLOSEFILE "marks.txt"
OUTPUT Count, " marks, mean ", Total / Count
```

常见错误

- 使用 `WHILE EOF(...)` 循环 — 仅在剩余内容已耗尽时执行一次。应改用 `NOT`。
- 因缺少 `EOF` 保护而越界读取。
- 打开一个从未被写入过的 `FOR READ` 文件。