文本文件

Ref As Pseudocode

打开与关闭文件

程序结束后，变量里的值就消失了。**文本文件**（text file）把数据保存在存储设备上，下次运行时还能再读出来。文本文件是由若干**行**组成的列表，每一行是一个字符串。各行按**顺序**一行接一行地读或写。

■ OPENFILE

使用文件之前，必须用某一种**文件模式**（file mode）把它打开：

```
OPENFILE <file identifier> FOR <file mode>
```

模式	作用
`READ`	从文件中读取各行，从第一行开始
`WRITE`	新建一个**空**文件并写入；文件中原有的数据会**丢失**
`APPEND`	在文件**末尾**添加新行；原有数据保留

文件标识符是文件名，可以是 `"Scores.txt"` 这样的字符串，也可以是存放文件名的 `STRING` 变量。

■ CLOSEFILE

```
CLOSEFILE <file identifier>
```

用完每个文件后都要关闭它。一个文件**同一时间只能用一种模式**打开：要读取刚写完的文件，先关闭它，再用 `FOR READ` 重新打开。

```
DECLARE FileName : STRING
FileName ← "Log.txt"
OPENFILE FileName FOR WRITE
WRITEFILE FileName, "Program started"
CLOSEFILE FileName
OUTPUT "Written to ", FileName
```

常见错误

- 没有先打开文件就使用它。
- 忘记写 `CLOSEFILE`，在 Paper 2 中这常常值一分。
- 想保留原有的行，却用 `WRITE` 打开。`WRITE` 会清空文件；应该用 `APPEND`。

写入与追加

WRITEFILE 向一个以 WRITE 或 APPEND 模式打开的文件写入**一行**。

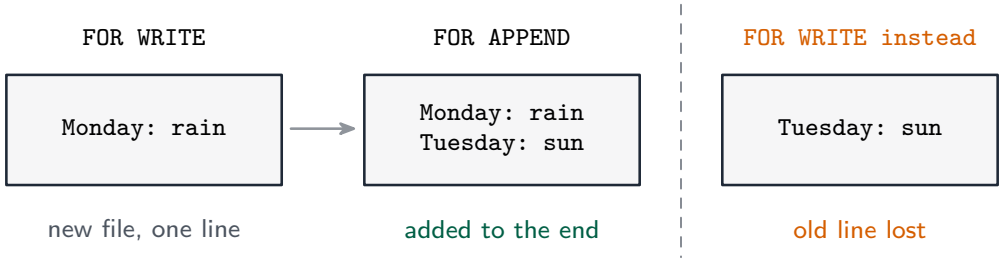
```
WRITEFILE <file identifier>, <data>
```

```
DECLARE Line : STRING
OPENFILE "Diary.txt" FOR WRITE
WRITEFILE "Diary.txt", "Monday: rain"
CLOSEFILE "Diary.txt"

OPENFILE "Diary.txt" FOR APPEND
WRITEFILE "Diary.txt", "Tuesday: sun"
CLOSEFILE "Diary.txt"

OPENFILE "Diary.txt" FOR READ
WHILE NOT EOF("Diary.txt")
   READFILE "Diary.txt", Line
   OUTPUT Line
ENDWHILE
CLOSEFILE "Diary.txt"
```

两行都会输出，因为 APPEND 保留了第一行。把 APPEND 改成 WRITE 再运行一次：只剩下星期二那一行。



WRITE 从空文件开始；*APPEND* 在末尾添加

一行存放多个值

一行就是一个字符串。要存放多个值，就用**分隔符**（separator），例如逗号，把它们连接成一个字符串：

```
DECLARE Name : STRING
DECLARE Score : INTEGER
Name ← "Ravi"
Score ← 87
OPENFILE "Results.txt" FOR WRITE
WRITEFILE "Results.txt", Name & "," & NUM_TO_STR(Score)
CLOSEFILE "Results.txt"
OUTPUT "Saved: ", Name & "," & NUM_TO_STR(Score)
```

保存整个数组

在循环中每行写入一个元素：

```
DECLARE Colours : ARRAY[1:3] OF STRING
DECLARE i : INTEGER
Colours[1] ← "red"
Colours[2] ← "green"
Colours[3] ← "blue"
OPENFILE "Colours.txt" FOR WRITE
FOR i ← 1 TO 3
   WRITEFILE "Colours.txt", Colours[i]
NEXT i
CLOSEFILE "Colours.txt"
OUTPUT "3 lines written"
```

把 OPENFILE 和 CLOSEFILE 放在循环**外面**。如果在循环里用 FOR WRITE 打开文件，每一遍都会把文件清空。

读取文件到末尾

READFILE 从一个以 FOR READ 模式打开的文件中读取**下一行**，并把它存进一个变量。这个变量应该是 STRING 类型。

```
READFILE <file identifier>, <variable>
```

函数 EOF 告诉你什么时候已经没有内容可读了。EOF 是**文件结束**（end of file）的缩写：

```
EOF(<file identifier>)
```

没有剩余的行时（包括文件为空时），EOF 返回 TRUE，否则返回 FALSE。

■ 标准的读文件循环

你通常不知道文件有多少行，而且它可能是空的。所以要用 WHILE 循环，在每次读取**之前**检查 EOF：

```
DECLARE Line : STRING
DECLARE Count : INTEGER
OPENFILE "Names.txt" FOR WRITE
WRITEFILE "Names.txt", "Amir"
WRITEFILE "Names.txt", ""
WRITEFILE "Names.txt", "Bea"
CLOSEFILE "Names.txt"

Count ← 0
OPENFILE "Names.txt" FOR READ
WHILE NOT EOF("Names.txt")
   READFILE "Names.txt", Line
   IF Line <> "" THEN
      Count ← Count + 1
      OUTPUT Line
   ENDIF
ENDWHILE
CLOSEFILE "Names.txt"
OUTPUT Count, " names read"
```

第一部分只是为了让例子能运行而创建文件。第二部分才是要记住的模板：**打开，当未到文件末尾时循环，读取，处理，关闭**。

■ 把一个文件复制到另一个文件

这是指南中的原例：复制每一行，但把空行换成一串短横线。

```
DECLARE LineOfText : STRING
OPENFILE "FileA.txt" FOR READ
OPENFILE "FileB.txt" FOR WRITE
WHILE NOT EOF("FileA.txt")
   READFILE "FileA.txt", LineOfText
   IF LineOfText = "" THEN
      WRITEFILE "FileB.txt", "-------------"
   ELSE
      WRITEFILE "FileB.txt", LineOfText
   ENDIF
ENDWHILE
CLOSEFILE "FileA.txt"
CLOSEFILE "FileB.txt"
```

两个文件可以同时打开，各用各的模式。

■ 练习

1. 文件 `Marks.txt` 每行存放一个整数分数。写出伪代码，读取这个文件并输出平均分。

<details> <summary> 参考答案 </summary>

```
DECLARE Line : STRING
DECLARE Total : INTEGER
DECLARE Count : INTEGER
// Create a sample file so that the program can run
OPENFILE "Marks.txt" FOR WRITE
WRITEFILE "Marks.txt", "60"
WRITEFILE "Marks.txt", "75"
WRITEFILE "Marks.txt", "90"
CLOSEFILE "Marks.txt"

Total ← 0
Count ← 0
OPENFILE "Marks.txt" FOR READ
WHILE NOT EOF("Marks.txt")
   READFILE "Marks.txt", Line
   Total ← Total + STR_TO_NUM(Line)
   Count ← Count + 1
ENDWHILE
CLOSEFILE "Marks.txt"
IF Count > 0 THEN
   OUTPUT "Average: ", Total / Count
ELSE
   OUTPUT "The file is empty"
ENDIF
```

每一行都是字符串，所以要用 `STR_TO_NUM` 把它变成数字。`IF Count > 0` 可以防止文件为空时出现除以零。

</details>

常见错误

- 用 `REPEAT … UNTIL EOF(…)`。它在检查**之前**就先读一次，遇到空文件会出错。
- 用 `FOR` 循环，并猜测文件的行数。
- 对读入的一行直接做算术，没有先转换。
- 读取以 `FOR WRITE` 打开的文件，或者向以 `FOR READ` 打开的文件写入。