OS Assignment4 Report

Chen Yian 118010030

October 2020

1 Introduction

Assignment 4 asked us to build one simplified version of file system. Here we need to implement three parts: Super Block, FCB, and file content area.

Considering that file content area shall be simply a storage and is mainly controlled by FCB area. We main skip the design of this part and mainly discuss about super block and FCB.

2 Program Design

2.1 Super Block

Super block area is one special area on the head of a storage device to represent which part is available. To use as least storage as possible, we use consecutive binary bits to represent the emptiness of storage disk's blocks. This is called the **Bit Map method**.

In this specific problem, we have 1024KB file-content area. With 32B as the size of one block, then we will have (1024KB/32B = **32K**) blocks. So we will need (32K bits = **4KB**) as super block area. This time, we store it as `char[4096]`, with each bytes recording 8 blocks' emptiness.

The operation of super block is closely related with defragmentation. We will leave this part in **2.4**.

2.2 File Control Block

File Control Block is a block corresponding with each file, recording each file's attributes, which includes: File name, file size, file address, file modification time. With 32B limit for each FCB, we need to pre-arrange memory for each attributes. The result will be:
File Name: **20B** FCB[0] - FCB[19]
File Size: **4B** FCB[20] - FCB[23]
File Address: **4B** FCB[24] - FCB[27]
File Modification Time: **4B** FCB[28] - FCB[31]

Surprisingly, the operation of FCB is also related with defragmentation. We will leave this part in **2.4**

2.3 Sorting

The sorting of file array requires several attributes due to the limited storage size and the rather small array size.

In this assignment, we only have 128B temp storage, thus only in-place sorting should be available. Also, by some trials, the author found out that no recursion sort are available with limited memory. Thus, the sorting algorithm left here is quite limited. The author applied choice sort.

Thanks to the limited array size, the sorting time consumption is not as long as expected. However, with larger files and more file occupied, it is recommended to apply a much better sorting algorithm to increase the sorting efficiency.

2.4 Defragmentation

Defragmentation is one important technique required in this assignment. As we applied contiguous allocation, we need to defragment every time we **removed** one file. In this context, we simply need to **maintain the linearity** of the file array.

This kind of defragmentaion simplifies our **super block modification**. Every time we delete one file with n blocks, we simply need to clear the last n bits of super block.

It is noted that, when sorting the FCB array, we also need to do defragmentation, this saves us a lot of times and trouble when sorting.

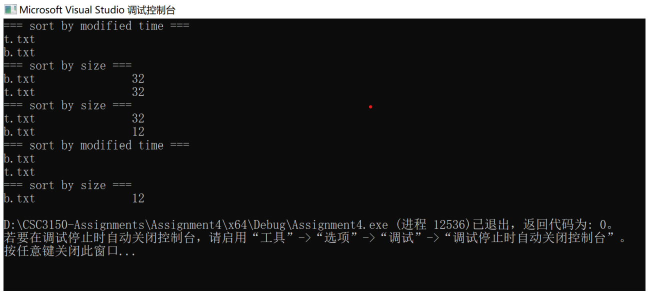
3 Environment

`Win 10 + cuda 10.0`

4 Execution Step

Click make.

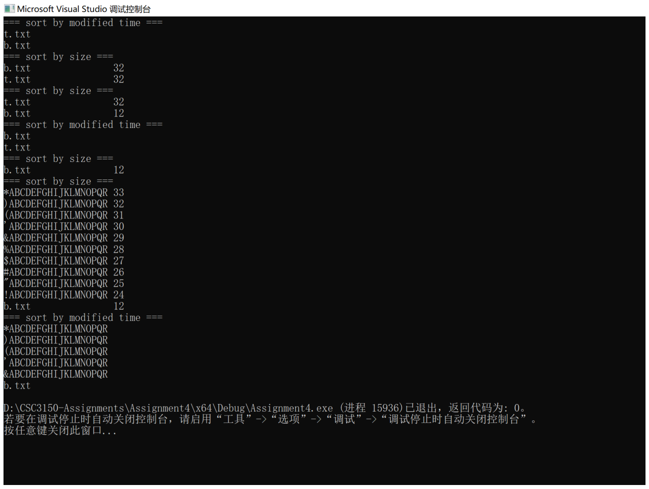
5 Result Display



```
=== sort by modified time ===
t.txt
b.txt
=== sort by size ===
b.txt               32
t.txt               32
=== sort by size ===
t.txt               32
b.txt               12
=== sort by modified time ===
b.txt
t.txt
=== sort by size ===
b.txt               12

D:\CSC3150-Assignments\Assignment4\x64\Debug\Assignment4.exe (进程 12536)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

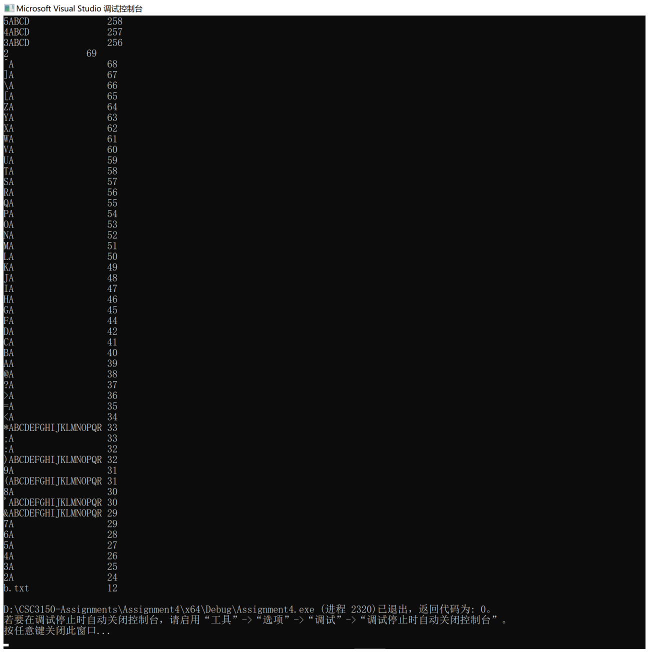
test case 1



```
=== sort by modified time ===
t.txt
b.txt
=== sort by size ===
b.txt               32
t.txt               32
=== sort by size ===
t.txt               32
b.txt               12
=== sort by modified time ===
b.txt
t.txt
=== sort by size ===
b.txt               12
=== sort by size ===
*ABCDEFGHIJKLMNOPQR 33
)ABCDEFGHIJKLMNOPQR 32
(ABCDEFGHIJKLMNOPQR 31
'ABCDEFGHIJKLMNOPQR 30
&ABCDEFGHIJKLMNOPQR 29
%ABCDEFGHIJKLMNOPQR 28
$ABCDEFGHIJKLMNOPQR 27
#ABCDEFGHIJKLMNOPQR 26
"ABCDEFGHIJKLMNOPQR 25
!ABCDEFGHIJKLMNOPQR 24
b.txt               12
=== sort by modified time ===
*ABCDEFGHIJKLMNOPQR
)ABCDEFGHIJKLMNOPQR
(ABCDEFGHIJKLMNOPQR
'ABCDEFGHIJKLMNOPQR
&ABCDEFGHIJKLMNOPQR
b.txt

D:\CSC3150-Assignments\Assignment4\x64\Debug\Assignment4.exe (进程 15936)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

test case 2



```
5ABCD               258
4ABCD               257
3ABCD               256
2               69
`A                  68
]A                  67
\A                  66
[A                  65
ZA                  64
YA                  63
XA                  62
WA                  61
VA                  60
UA                  59
TA                  58
SA                  57
RA                  56
QA                  55
PA                  54
OA                  53
NA                  52
MA                  51
LA                  50
KA                  49
JA                  48
IA                  47
HA                  46
GA                  45
FA                  44
DA                  42
CA                  41
BA                  40
AA                  39
@A                  38
?A                  37
>A                  36
=A                  35
<A                  34
*ABCDEFGHIJKLMNOPQR 33
;A                  33
:A                  32
)ABCDEFGHIJKLMNOPQR 32
9A                  31
(ABCDEFGHIJKLMNOPQR 31
8A                  30
'ABCDEFGHIJKLMNOPQR 30
&ABCDEFGHIJKLMNOPQR 29
7A                  29
6A                  28
5A                  27
4A                  26
3A                  25
2A                  24
b.txt               12

D:\CSC3150-Assignments\Assignment4\x64\Debug\Assignment4.exe (进程 2320)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

test case 3