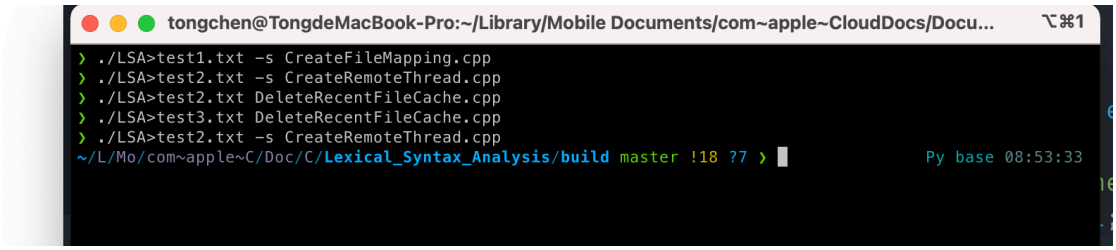
词法分析测试报告

陈童
2019211283
304班



通过如下命令，对CreateFileMapping.cpp，CreateRemoteThread.cpp，DeleteRecentFileCache.cpp三个文件（随机选自Github代码库，来源https://github.com/3gstudent/Homework-of-C-Language），生成test1.txt, test2.txt, test3.txt三个输出文件。下对此进行分析。

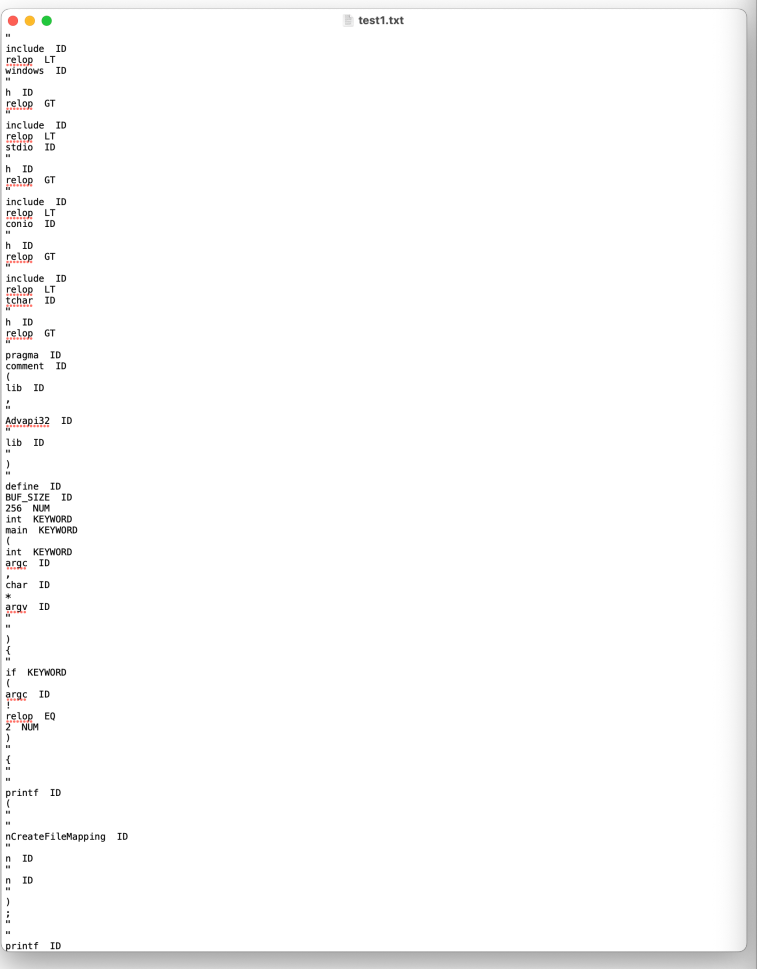
CreateFileMapping.cpp

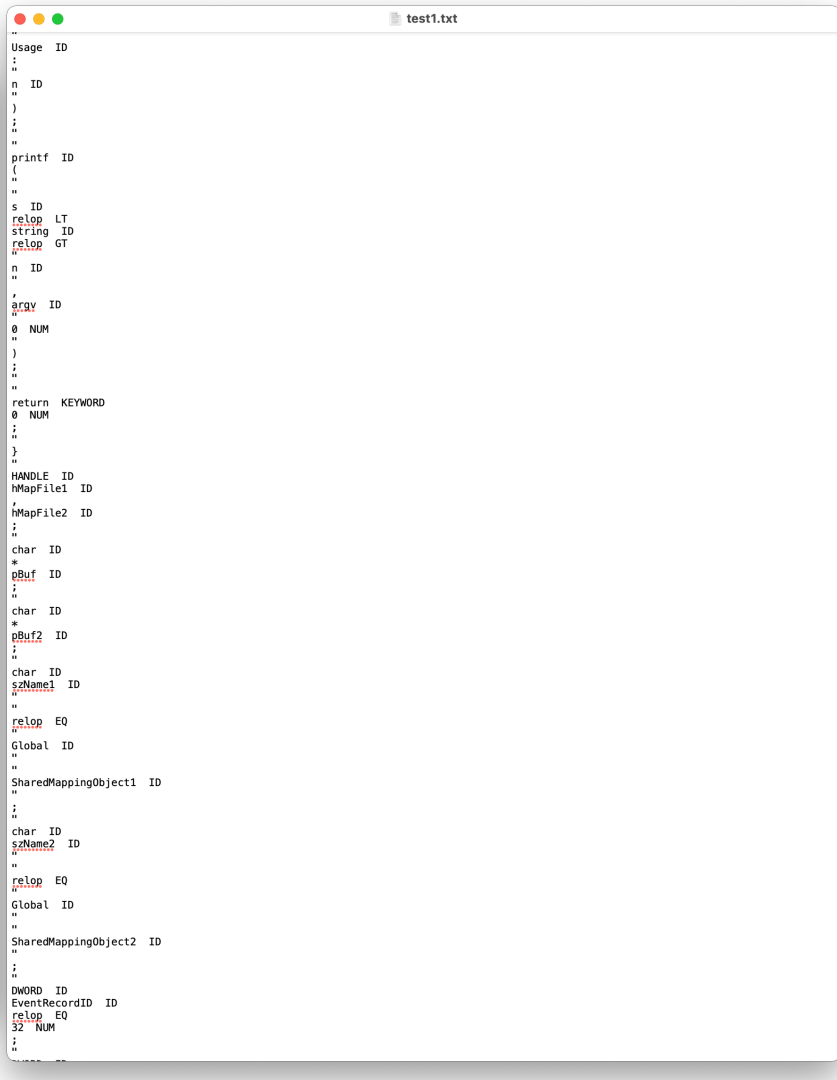
CreateFileMapping.cpp 9+, U

Lexical_Syntax_Analysis > build > CreateFileMapping.cpp > main

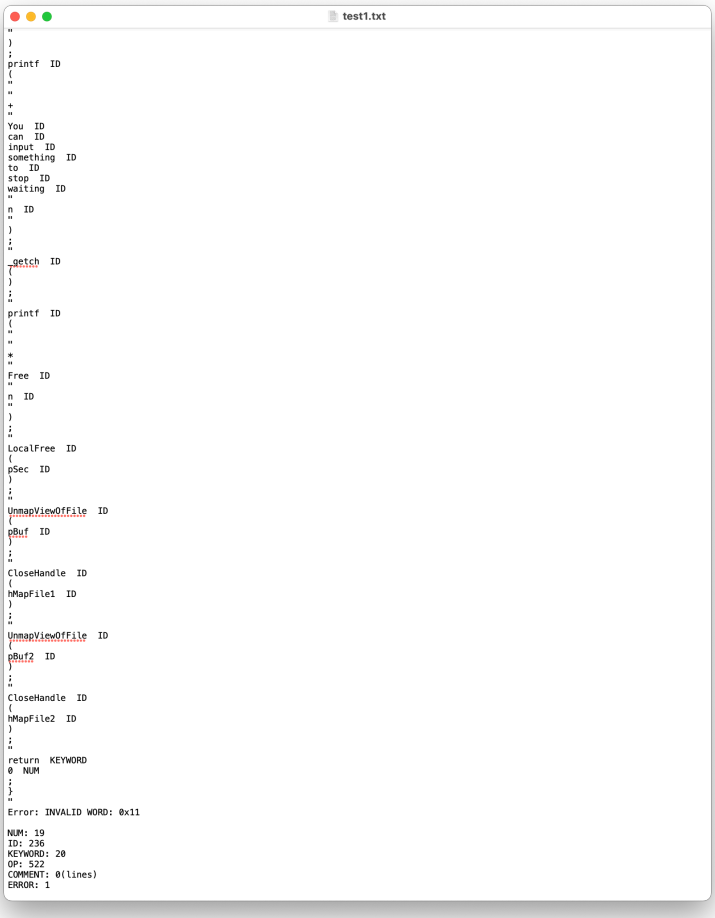
```cpp
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include <tchar.h>
#pragma comment(lib,"Advapi32.lib")

#define BUF_SIZE 256

int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        printf("\nCreateFileMapping\n\n");
        printf("Usage:\n");
        printf("%s <string>\n",argv[0]);
        return 0;
    }

    HANDLE hMapFile1, hMapFile2;
    char *pBuf;
    char *pBuf2;
    char szName1[] = "Global\\SharedMappingObject1";
    char szName2[] = "Global\\SharedMappingObject2";
    DWORD EventRecordID = 32;
    DWORD offset = 0x11;

    char szOffset[8];
    sprintf_s(szOffset, "%d", offset);

    printf("[*]Try to set SecurityDescriptor... ");

    PSECURITY_DESCRIPTOR pSec = (PSECURITY_DESCRIPTOR)LocalAlloc(LMEM_FIXED, SECURITY_DESCRIPTOR_MIN_LENGTH);
    if (!pSec)
    {
        return GetLastError();
    }
    if (!InitializeSecurityDescriptor(pSec, SECURITY_DESCRIPTOR_REVISION))
    {
        LocalFree(pSec);
        return GetLastError();
```

通过LSA输出为





其中可以看出，输出形式为二元表<标记，属性>，每个输出均符合翻译表规范。在文件最后输出了统计信息。可以发现该LSA不支持对十六进制数和八进制数的分析读入。



在对应代码块和输出部分可看出，LSA实现了一定的错误处理能力

```
;
"
DWORD  ID
EventRecordID  ID
relop  EQ
32  NUM
;
"
DWORD  ID
offset  ID
relop  EQ
;
"
char  ID
szOffset  ID
"
8  NUM
"
;
"
sprintf_s  ID
```

```
DWORD EventRecordID = 32;
DWORD offset = 0x11;
```

CreateRemoteThread.cpp

```
CreateRemoteThread.cpp 9+, U
Lexical_Syntax_Analysis > build > CreateRemoteThread.cpp > ...
                                                                    0x
1   #include <windows.h>
2   #include <stdio.h>
3   #include <tlhelp32.h>
4   #pragma comment(lib,"Advapi32.lib")
5
6   BOOL InjectDll(UINT32 ProcessId, char *DllPath)
7   {
8       if (strstr(DllPath, "\\\\") != 0)
9       {
10          printf("[!]Wrong Dll path\n");
11          return FALSE;
12      }
13      if (strstr(DllPath, "\\") == 0)
14      {
15          printf("[!]Need Dll full path\n");
16          return FALSE;
17      }
18
19      size_t len = strlen(DllPath) + 1;
20
21      LPVOID pThreadData = NULL;
22      HANDLE ProcessHandle = NULL;
23      HANDLE hThread = NULL;
24      BOOL bRet = FALSE;
25
26      __try
27      {
28          ProcessHandle = OpenProcess(PROCESS_ALL_ACCESS, FALSE, ProcessId);
29          if (ProcessHandle == NULL)
30          {
31              printf("[!]OpenProcess error\n");
32              __leave;
33          }
```

DeleteRecentFileCache.cpp

Lexical_Syntax_Analysis > build > DeleteRecentFileCache.cpp > DeleteRecord

```cpp
#include <windows.h>
#pragma pack(1)

typedef struct _BCF_HEADER {
    ULONG64 Flag1;
    ULONG64 Flag2;
    ULONG Unknown;
} BCFHEADER, *PBCFHEADER;

typedef struct _BCF_RECORD {
    ULONG Size;
} BCFRECORD, *PBCFRECORD;
#pragma pack()

int NewSize = 0;

char *DeleteRecord(PVOID mapAddress, char *TempBuf, int StopSize,WCHAR *FileName)
{
    char flag[16] = { 0xFE,0xFF,0xEE,0xFF,0x11,0x22,0x00,0x00,0x03,0x00,0x00,0x00,0x01,0x00,0x
    if (!memcmp(mapAddress, flag, 16))
    {
        printf("[!]Maybe it's not RecentFileCache.bcf");
        exit(0);
    }
    memcpy(TempBuf, mapAddress, 0x14);
    PBCFRECORD currentRecordPtr = NULL;
    PBCFRECORD nextRecordPtr = (PBCFRECORD)((PBYTE)mapAddress + 0x14);
    int DeleteSize = 0;
    int FlagSize = 0x14;
    while (FlagSize + DeleteSize < StopSize)
    {
        currentRecordPtr = nextRecordPtr;

        WCHAR *RecordName = new WCHAR[nextRecordPtr->Size + 1];
        memcpy(RecordName, nextRecordPtr + 1, nextRecordPtr->Size * 2 + 2);
        printf("%ws\n", RecordName);
        if (wcscmp(RecordName, FileName) == 0)
```

0x test3.txt U

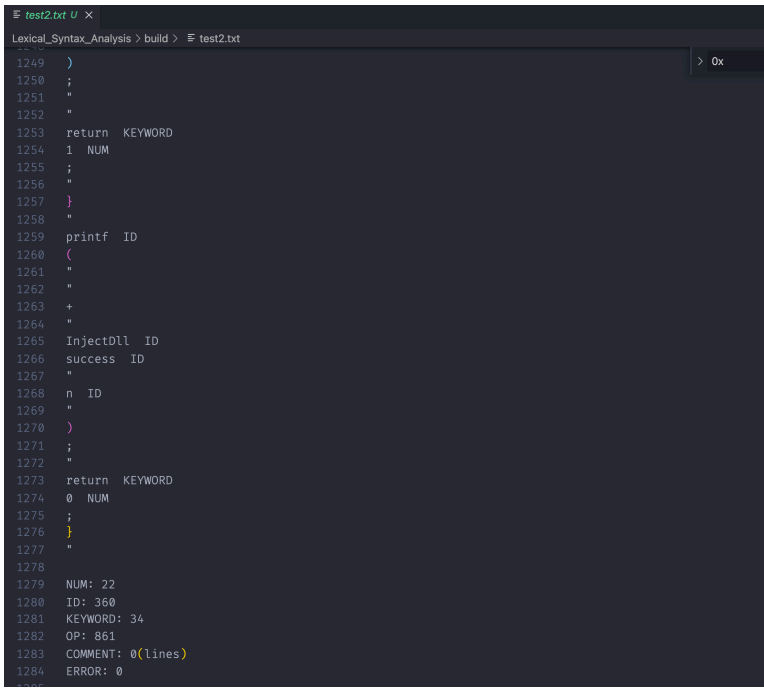
Lexical_Syntax_Analysis > build > test3.txt

```
"
include  ID
relop  LT
windows  ID
"
h  ID
relop  GT
"
pragma  ID
pack  ID
(
1  NUM
)
typedef  ID
struct  ID
_BCF_HEADER  ID
{
"
ULONG64  ID
Flag1  ID
;
"
ULONG64  ID
Flag2  ID
;
"
ULONG  ID
Unknown  ID
;
}
BCFHEADER  ID
,
*
PBCFHEADER  ID
;
typedef  ID
```

通过错误分析可看出，LSA对字符串支持的不是很好。

```
f("The new file will be saved as NewRec
f("Author:3gstudent\n");
f("Usage:\n");
f("      %s <file path of RecentFileCach
f("eg:\n");
f("      %s C:\\Windows\\AppCompat\\Prog
f("[!]Wrong parameter\n");
n 0;


 fp;
rr = fopen_s(&fp, argv[1], "a+");
```

```
Error: INVALID WORD: 3gstudent
```


test2.txt U
Lexical_Syntax_Analysis > build > test2.txt
0x
)
;
"
"
return KEYWORD
1 NUM
;
"
}
"
printf ID
(
"
"
+
"
InjectDll ID
success ID
"
n ID
"
)
;
"
return KEYWORD
0 NUM
;
}
"
NUM: 22
ID: 360
KEYWORD: 34
OP: 861
COMMENT: 0(lines)
ERROR: 0