

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 #include<string.h>
4 #include"linked_list.h"
5
6 //存放指令和對應opcode的結構，用來查詢比對
7 typedef struct opcode{
8     char *command;
9     int code;
10 }opcode;
11
12 //總共有26個指令
13 opcode sic[26];
14
15 //紀錄掃描到的標籤及地址的單向鏈結串列的節點
16 typedef struct label{
17     char *label_name;
18     int address;
19     llNode_t node;
20 }label;
21
22 //暫存object code的結構(使用位域)
23 typedef struct sic24_t{
24     unsigned int address : 16;
25     unsigned int opcode : 8;
26 }sic24_t;
27
28 //顯示一個單向鏈結串列所有節點的資料(用在開發中途測試串列是否正常運作)
29 int printf_all_list(llNode_t *head)
30 {
31     llNode_t *current = head;
32     while(current->next != NULL )
33     {
34         current = current->next;
35         printf("%-8s ", return_to_user_struct_pointer(label, node, current)->label_name);
36         printf("%X\n", return_to_user_struct_pointer(label, node, current)->address);
37     }
38     printf("end\n");
39     return 0;
40 }
41
42
43
44 //將指數乘開的函數
45 int exponent_Int(const int base, int n)
46 {
47     int p = base;
48     if(n == 0)
49     {
50         p = 1;
51     }
52     else
53     {
54         for (int i = 1; i < n; i++)
55         {
56             p *= base;
57         }
58     }
```

```
59     return p;
60 }
61
62 //將讀取的字串視為16進位，並轉換為10進位數字的函數
63 int hex_to_dex(char *hex)
64 {
65     char *char_temp = (char *)malloc(strlen(hex));
66     strcpy(char_temp, hex);
67     char_temp = strtok(char_temp, "\\n");
68     char temp[2];
69     int total = 0;
70     int count = strlen(char_temp);
71
72     while(count-- && count >= 0)
73     {
74         sprintf(temp, "%c", *char_temp);
75         if(strcmp(temp, "A") == 0){total += exponent_Int(16, count)*10;}
76         if(strcmp(temp, "B") == 0){total += exponent_Int(16, count)*11;}
77         if(strcmp(temp, "C") == 0){total += exponent_Int(16, count)*12;}
78         if(strcmp(temp, "D") == 0){total += exponent_Int(16, count)*13;}
79         if(strcmp(temp, "E") == 0){total += exponent_Int(16, count)*14;}
80         if(strcmp(temp, "F") == 0){total += exponent_Int(16, count)*15;}
81         else{total += exponent_Int(16, count)*atoi(temp);}
82         char_temp++;
83     }
84
85     return total;
86 }
87
88 //取得命令中的目標檔名做為參數
89 int main(int argc, char *argv[])
90 {
91     //初始化每種指令及其對應的opcode
92     sic[0].command = "ADD";
93     sic[0].code = 24;
94
95     sic[1].command = "DIV";
96     sic[1].code = 36;
97
98     sic[2].command = "JGT";
99     sic[2].code = 52;
100
101     sic[3].command = "LDA";
102     sic[3].code = 0;
103
104     sic[4].command = "LDX";
105     sic[4].code = 4;
106
107     sic[5].command = "RD";
108     sic[5].code = 216;
109
110     sic[6].command = "STCH";
111     sic[6].code = 84;
112
113     sic[7].command = "STX";
114     sic[7].code = 16;
115
116     sic[8].command = "TIX";
117     sic[8].code = 44;
118 }
```

```
119     sic[9].command = "AND";
120     sic[9].code = 64;
121
122     sic[10].command = "J";
123     sic[10].code = 60;
124
125     sic[11].command = "JLT";
126     sic[11].code = 56;
127
128     sic[12].command = "LDCH";
129     sic[12].code = 80;
130
131     sic[13].command = "MUL";
132     sic[13].code = 32;
133
134     sic[14].command = "RSUB";
135     sic[14].code = 76;
136
137     sic[15].command = "STL";
138     sic[15].code = 20;
139
140     sic[16].command = "SUB";
141     sic[16].code = 28;
142
143     sic[17].command = "WD";
144     sic[17].code = 220;
145
146     sic[18].command = "COMP";
147     sic[18].code = 40;
148
149     sic[19].command = "JEQ";
150     sic[19].code = 48;
151
152     sic[20].command = "JSUB";
153     sic[20].code = 72;
154
155     sic[21].command = "LDL";
156     sic[21].code = 8;
157
158     sic[22].command = "OR";
159     sic[22].code = 68;
160
161     sic[23].command = "STA";
162     sic[23].code = 12;
163
164     sic[24].command = "STSW";
165     sic[24].code = 232;
166
167     sic[25].command = "TD";
168     sic[25].code = 224;
169
170     //***** 第一階段 檔案處理
171     //*****
172
173     for (int i = 1; i < argc; i++)
174     {
175         //打開來源檔案，同時組合新的檔名並以它新增空白檔案
176         FILE *ass = fopen(argv[i], "r");
177         printf("[%d] [source] %s\n", i, argv[i]);
```

```
178 char *obj_file_name = strtok(argv[i], ".");
179 char *obj= ".o";
180 strcat(obj_file_name, obj);
181 FILE *obj_file = fopen(obj_file_name, "w");
182 printf("[%d] [target] %s\n\n", i, obj_file_name);
183
184
185 //宣告組譯過程中會用到的buffer及變數
186 char buffer[50];
187 char obj_temp[100];
188 char obj_cat_temp[60];
189 char record_head_temp[50];
190 char temp1[50];
191 char temp2[50];
192 char temp3[50];
193 char temp4[50];
194 char temp5[50];
195 int code_address = 0;
196 llNode_t *label_list = LL_init();
197 llNode_t *label_list_temp;
198 label *new = NULL;
199 char program_name[10];
200 int START_address = 0;
201 int END_address = 0;
202 sic24_t obj_code;
203 char *BYTE_temp;
204 int arg_get = 0;
205 int T_count = 0;
206 int T_start_count = 0;
207 llNode_t *free_temp;
208 label *free_top_temp;
209
210
211 //第一次掃描開始，針對不同的狀況及規則算出地址
212 while(fgets(buffer, 50, ass))
213 {
214     memset(temp1, 0, 50);
215     memset(temp2, 0, 50);
216     memset(temp3, 0, 50);
217     memset(obj_cat_temp, 0, 50);
218
219     sscanf(buffer, "%s %s %s", temp1, temp2, temp3);
220     if(strcmp(temp1, ".") != 0 )
221     {
222         if(strcmp(temp2, "START") != 0)
223         {
224             if(strcmp(temp1, "END") != 0)
225             {
226                 for(int i = 0; i < 26; i++)
227                 {
228                     if(strcmp(temp1, sic[i].command) == 0)
229                     {
230                         code_address+=3;
231                     }
232                 }
233             }
234
235             for(int i = 0; i < 26; i++)
236             {
237                 if(strcmp(temp2, sic[i].command) == 0)
```

```
238     {
239         //如果是標籤就在串列尾端新增一個節點紀錄標籤名及地址
240         new = malloc(sizeof(label));
241         new->label_name = malloc(strlen(temp1));
242         strcpy(new->label_name, temp1);
243         new->address = code_address;
244         LL_add_tail(&new->node, label_list);
245
246         code_address+=3;
247     }
248 }
249
250 if(strcmp(temp2, "RESB") == 0)
251 {
252     //如果是標籤就在串列尾端新增一個節點紀錄標籤名及地址
253     new = malloc(sizeof(label));
254     new->label_name = malloc(strlen(temp1));
255     strcpy(new->label_name, temp1);
256     new->address = code_address;
257     LL_add_tail(&new->node, label_list);
258
259     code_address += atoi(temp3);
260 }
261
262 if(strcmp(temp2, "RESW") == 0)
263 {
264     //如果是標籤就在串列尾端新增一個節點紀錄標籤名及地址
265     new = malloc(sizeof(label));
266     new->label_name = malloc(strlen(temp1));
267     strcpy(new->label_name, temp1);
268     new->address = code_address;
269     LL_add_tail(&new->node, label_list);
270
271     code_address += 3*atoi(temp3);
272 }
273
274 if(strcmp(temp2, "BYTE") == 0)
275 {
276     //如果是標籤就在串列尾端新增一個節點紀錄標籤名及地址
277     new = malloc(sizeof(label));
278     new->label_name = malloc(strlen(temp1));
279     strcpy(new->label_name, temp1);
280     new->address = code_address;
281     LL_add_tail(&new->node, label_list);
282
283     sscanf(temp3, "%[^']%[^']", temp4, temp5);
284
285     if(*temp4 == 'C')
286     {
287         code_address += strlen(temp5);
288     }
289
290     if(*temp4 == 'X')
291     {
292         code_address += strlen(temp5)/2 + strlen(temp5)%2;
293     }
294 }
295
296
297 if(strcmp(temp2, "WORD") == 0)
```

```

298         {
299             //如果是標籤就在串列尾端新增一個節點紀錄標籤名及地址
300             new = malloc(sizeof(label));
301             new->label_name = malloc(strlen(temp1));
302             strcpy(new->label_name, temp1);
303             new->address = code_address;
304             LL_add_tail(&new->node, label_list);
305
306             code_address += 3;
307         }
308     }
309     else
310     {
311         END_address = code_address;
312     }
313 }
314 }
315 else
316 {
317     code_address += hex_to_dex(temp3);
318     START_address = code_address;
319     strcpy(program_name, temp1);
320 }
321 }
322
323 }
324
325 //印出object file 的檔頭片段
326 printf("H^%-6s^%06X^%06X\n", program_name, START_address, END_address -
START_address);
327 sprintf(record_head_temp, "H^%-6s^%06X^%06X\n", program_name, START_address,
END_address - START_address);
328 fputs(record_head_temp, obj_file);
329
330 //重置檔案資料流指針回起點
331 rewind(ass);
332 code_address = 0;
333
334 //第二次掃描開始
335 while(fgets(buffer, 50, ass))
336 {
337
338     memset(temp1, 0, 50);
339     memset(temp2, 0, 50);
340     memset(temp3, 0, 50);
341
342
343     sscanf(buffer, "%s %s %s", temp1, temp2, temp3);
344     if(strcmp(temp1, ".") != 0 )
345     {
346         if(strcmp(temp2, "START") != 0)
347         {
348             if(strcmp(temp1, "END") != 0)
349             {
350                 for(int i = 0; i < 26; i++)
351                 {
352                     //比對第一段字串
353                     if(strcmp(temp1, sic[i].command) == 0)
354                     {
355                         obj_code.opcode = sic[i].code;

```

```

356         label_list_temp = label_list;
357
358         while(label_list_temp->next != NULL)
359         {
360             label_list_temp = LL_next_node(label_list_temp);
361             if(strcmp(temp2,
return_to_user_struct_pointer(label, node, label_list_temp)->label_name) == 0)
362             {
363                 obj_code.addres =
return_to_user_struct_pointer(label, node, label_list_temp)->addres;
364                 arg_get = 1;
365             }
366         }
367
368         if(strcmp(temp1, "RSUB") != 0 && arg_get == 0)
369         {
370             sscanf(temp2, "%[^,],%s", temp4, temp5);
371
372             label_list_temp = label_list;
373             //比對參數是否為標籤
374             while(label_list_temp->next != NULL)
375             {
376                 label_list_temp =
LL_next_node(label_list_temp);
377                 if(strcmp(temp4,
return_to_user_struct_pointer(label, node, label_list_temp)->label_name) == 0)
378                 {
379                     obj_code.addres =
return_to_user_struct_pointer(label, node, label_list_temp)->addres;
380                 }
381             }
382
383             if(*temp5 == 'X')
384             {
385                 obj_code.addres += 32768;
386             }
387         }
388     }
389
390     //如果片段加上這次的object code會超過30個Byte
391     if(T_count + 3 > 30)
392     {
393         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
394         息寫進檔案中一行的起頭
395         printf("T^%06X^%02X", T_start_count, T_count);
396         sprintf(record_head_temp, "T^%06X^%02X",
T_start_count, T_count);
397         fputs(record_head_temp, obj_file);
398
399         //輸出整段object code 到終端機和檔案
400         printf("%s\n", obj_temp);
401         fputs(obj_temp, obj_file);
402         fputs("\n", obj_file);
403         memset(obj_temp, 0, 100);
404         T_count = 0;
405     }
406
407     if(T_count == 0)
408     {

```

```

409         sprintf(obj_temp, "%02X%04X", obj_code.opcode,
obj_code.address);
410         T_start_count = code_address;
411     }
412     else
413     {
414         sprintf(obj_cat_temp, "%02X%04X",
obj_code.opcode, obj_code.address);
415         strcat(obj_temp, obj_cat_temp);
416     }
417
418     T_count += 3;
419
420     //如果片段剛好裝滿30個Byte
421     if(T_count == 30)
422     {
423         //顯示片段起始訊息，同時在輸出片段到檔案前先把起始訊
息寫進檔案中一行的起頭
424         printf("T^%06X^%02X", T_start_count, T_count);
425         sprintf(record_head_temp, "T^%06X^%02X",
T_start_count, T_count);
426         fputs(record_head_temp, obj_file);
427
428         //輸出整段object code 到終端機和檔案
429         printf("%s\n", obj_temp);
430         fputs(obj_temp, obj_file);
431         fputs("\n", obj_file);
432         memset(obj_temp, 0, 100);
433
434         T_count = 0;
435     }
436
437     code_address+=3;
438     obj_code.opcode = 0;
439     obj_code.address = 0;
440 }
441
442 //比對第二段字串
443 for(int i = 0; i < 26; i++)
444 {
445     if(strcmp(temp2, sic[i].command) == 0)
446     {
447
448         obj_code.opcode = sic[i].code;
449
450         label_list_temp = label_list;
451         while(label_list_temp->next != NULL)
452         {
453             label_list_temp = LL_next_node(label_list_temp);
454             if(strcmp(temp3,
return_to_user_struct_pointer(label, node, label_list_temp)->label_name) == 0)
455             {
456                 obj_code.address =
return_to_user_struct_pointer(label, node, label_list_temp)->address;
457             }
458         }
459
460         //如果片段加上這次的object code會超過30個Byte
461

```



```

462 if(T_count + 3 > 30)
463 {
464     //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
    息寫進檔案中一行的起頭
465     printf("T^%06X^%02X", T_start_count, T_count);
466     sprintf(record_head_temp, "T^%06X^%02X",
    T_start_count, T_count);
467     fputs(record_head_temp, obj_file);
468
469     //輸出整段object code 到終端機和檔案
470     printf("%s\n", obj_temp);
471     fputs(obj_temp, obj_file);
472     fputs("\n", obj_file);
473     memset(obj_temp, 0, 100);
474     T_count = 0;
475 }
476
477 if(T_count == 0)
478 {
479     T_start_count = code_address;
480     sprintf(obj_temp, "%02X%04X", obj_code.opcode,
    obj_code.addres);
481 }
482 else
483 {
484     sprintf(obj_cat_temp, "%02X%04X",
    obj_code.opcode, obj_code.addres);
485     strcat(obj_temp, obj_cat_temp);
486 }
487
488 T_count += 3;
489
490 //如果片段剛好裝滿30個Byte
491 if(T_count == 30)
492 {
493     //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
    息寫進檔案中一行的起頭
494     printf("T^%06X^%02X", T_start_count, T_count);
495     sprintf(record_head_temp, "T^%06X^%02X",
    T_start_count, T_count);
496     fputs(record_head_temp, obj_file);
497
498     //輸出整段object code 到終端機和檔案
499     printf("%s\n", obj_temp);
500     fputs(obj_temp, obj_file);
501     fputs("\n", obj_file);
502     memset(obj_temp, 0, 100);
503
504     T_count = 0;
505 }
506
507 code_address+=3;
508 obj_code.opcode = 0;
509 obj_code.addres = 0;
510 }
511
512 //命令以外的其他狀況處理
513 if(strcmp(temp2, "RESB") == 0)
514 {

```

```

515
516 //如果片段加上這次的object code會超過30個Byte
517 if(T_count + atoi(temp3) > 30)
518 {
519     //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊息寫進
    檔案中一行的起頭
520     printf("T^%06X^%02X", T_start_count, T_count);
521     sprintf(record_head_temp, "T^%06X^%02X",
    T_start_count, T_count);
522     fputs(record_head_temp, obj_file);
523
524     //輸出整段object code 到終端機和檔案
525     printf("%s\n", obj_temp);
526     fputs(obj_temp, obj_file);
527     fputs("\n", obj_file);
528     memset(obj_temp, 0, 100);
529     T_count = 0;
530 }
531
532 //如果片段剛好裝滿30個Byte
533 if(T_count == 30)
534 {
535     //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊息寫進
    檔案中一行的起頭
536     printf("T^%06X^%02X", T_start_count, T_count);
537     sprintf(record_head_temp, "T^%06X^%02X",
    T_start_count, T_count);
538     fputs(record_head_temp, obj_file);
539
540     //輸出整段object code 到終端機和檔案
541     printf("%s", obj_temp);
542     fputs(obj_temp, obj_file);
543     fputs("\n", obj_file);
544     memset(obj_temp, 0, 100);
545
546     T_count = 0;
547 }
548 code_address += atoi(temp3);
549 }
550
551 if(strcmp(temp2, "RESW") == 0)
552 {
553     code_address += 3*atoi(temp3);
554 }
555
556 if(strcmp(temp2, "BYTE") == 0)
557 {
558     sscanf(temp3, "%[^']%[^']", temp4, temp5);
559
560     if(*temp4 == 'C')
561     {
562         //如果片段加上這次的object code會超過30個Byte
563         if(T_count + strlen(temp5) > 30)
564         {
565             //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
            息寫進檔案中一行的起頭
566             printf("T^%06X^%02X", T_start_count, T_count);
567             sprintf(record_head_temp, "T^%06X^%02X",
            T_start_count, T_count);

```

```

568         fputs(record_head_temp, obj_file);
569
570         //輸出整段object code 到終端機和檔案
571         printf("%s\n", obj_temp);
572         fputs(obj_temp, obj_file);
573         fputs("\n", obj_file);
574         memset(obj_temp, 0, 100);
575         T_count = 0;
576     }
577
578     if(T_count == 0)
579     {
580         T_start_count = code_address;
581         strcat(obj_temp, "^");
582
583         BYTE_temp = temp5;
584         for(int i = 0 ; i < strlen(temp5) ; i++)
585         {
586             sprintf(obj_cat_temp, "%X" , *BYTE_temp);
587             strcat(obj_temp, obj_cat_temp);
588             BYTE_temp++;
589         }
590     }
591     else
592     {
593         strcat(obj_temp, "^");
594         BYTE_temp = temp5;
595         for(int i = 0 ; i < strlen(temp5) ; i++)
596         {
597             sprintf(obj_cat_temp, "%X" , *BYTE_temp);
598             strcat(obj_temp, obj_cat_temp);
599             BYTE_temp++;
600         }
601     }
602
603     T_count += strlen(temp5);
604
605     //如果片段剛好裝滿30個Byte
606     if(T_count == 30)
607     {
608         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
        息寫進檔案中一行的起頭
609
610         printf("T^%06X^%02X", T_start_count, T_count);
611         sprintf(record_head_temp, "T^%06X^%02X",
        T_start_count, T_count);
612
613         fputs(record_head_temp, obj_file);
614
615         //輸出整段object code 到終端機和檔案
616         printf("%s\n", obj_temp);
617         fputs(obj_temp, obj_file);
618         fputs("\n", obj_file);
619         memset(obj_temp, 0, 100);
620
621         T_count = 0;
622     }
623     code_address += strlen(temp5);
624 }
625
626 if(*temp4 == 'X')

```

```

624 {
625     //如果片段加上這次的object code會超過30個Byte
626     if(T_count + strlen(temp5)/2 + strlen(temp5)%2 > 30)
627     {
628         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
        息寫進檔案中一行的起頭
629         printf("T^%06X^%02X", T_start_count, T_count);
630         sprintf(record_head_temp, "T^%06X^%02X",
        T_start_count, T_count);
631         fputs(record_head_temp, obj_file);
632
633         //輸出整段object code 到終端機和檔案
634         printf("%s\n", obj_temp);
635         fputs(obj_temp, obj_file);
636         fputs("\n", obj_file);
637         memset(obj_temp, 0, 100);
638         T_count = 0;
639     }
640
641     if(T_count == 0)
642     {
643         T_start_count = code_address;
644
645         sprintf(obj_temp, "^%s", temp5);
646     }
647     else
648     {
649         sprintf(obj_cat_temp, "%s", temp5);
650         strcat(obj_temp, obj_cat_temp);
651     }
652     T_count += strlen(temp5)/2 + strlen(temp5)%2;
653
654     //如果片段剛好裝滿30個Byte
655     if(T_count == 30)
656     {
657         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊
        息寫進檔案中一行的起頭
658         printf("T^%06X^%02X", T_start_count, T_count);
659         sprintf(record_head_temp, "T^%06X^%02X",
        T_start_count, T_count);
660         fputs(record_head_temp, obj_file);
661
662         //輸出整段object code 到終端機和檔案
663         printf("%s\n", obj_temp);
664         fputs(obj_temp, obj_file);
665         fputs("\n", obj_file);
666         memset(obj_temp, 0, 100);
667         T_count = 0;
668     }
669     code_address += strlen(temp5)/2 + strlen(temp5)%2;
670 }
671
672 }
673
674 if(strcmp(temp2, "WORD") == 0)
675 {
676     //如果片段加上這次的object code會超過30個Byte
677     if(T_count + 3 > 30)

```

```

678     {
679         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊息寫進
        檔案中一行的起頭
680         printf("T^%06X^%02X", T_start_count, T_count);
681         sprintf(record_head_temp, "T^%06X^%02X",
        T_start_count, T_count);
682         fputs(record_head_temp, obj_file);
683
684         //輸出整段object code 到終端機和檔案
685         printf("%s\n", obj_temp);
686         fputs(obj_temp, obj_file);
687         fputs("\n", obj_file);
688         memset(obj_temp, 0, 100);
689         T_count = 0;
690     }
691
692     if(T_count == 0)
693     {
694         T_start_count = code_address;
695         sprintf(obj_temp, "^%06X", atoi(temp3));
696     }
697     else
698     {
699         sprintf(obj_cat_temp, "^%06X", atoi(temp3));
700         strcat(obj_temp, obj_cat_temp);
701     }
702
703     T_count += 3;
704
705     //如果片段剛好裝滿30個Byte
706     if(T_count == 30)
707     {
708         //顯示片段起始訊息・同時在輸出片段到檔案前先把起始訊息寫進
        檔案中一行的起頭
709         printf("T^%06X^%02X", T_start_count, T_count);
710         sprintf(record_head_temp, "T^%06X^%02X",
        T_start_count, T_count);
711         fputs(record_head_temp, obj_file);
712
713         //輸出整段object code 到終端機和檔案
714         printf("%s\n", obj_temp);
715         fputs(obj_temp, obj_file);
716         fputs("\n", obj_file);
717         memset(obj_temp, 0, 100);
718
719         T_count = 0;
720     }
721     code_address += 3;
722 }
723 }
724 else
725 {
726     code_address += hex_to_dex(temp3);
727 }
728 }
729 arg_get = 0;
730 }
731
732 //如果翻譯完有剩餘的片段・把剩下的印出來

```

```
733     if(strcmp(obj_temp, "") != 0)
734     {
735         printf("T^%06X^%02X", T_start_count, T_count);
736         sprintf(record_head_temp, "T^%06X^%02X", T_start_count, T_count);
737         fputs(record_head_temp, obj_file);
738
739         printf("%s\n", obj_temp);
740         fputs(obj_temp, obj_file);
741         fputs("\n", obj_file);
742     }
743
744     //印出object file 的終止片段
745     printf("E^%06X\n", START_address);
746     sprintf(record_head_temp, "E^%06X", START_address);
747     fputs(record_head_temp, obj_file);
748
749     //關閉來源檔案及目標檔案
750     fclose(ass);
751     fclose(obj_file);
752
753     //釋放紀錄標籤的鏈結串列
754     while(!LL_isEmpty(label_list))
755     {
756         free_temp = LL_next_node(label_list);
757         free_top_temp = return_to_user_struct_pointer(label, node, free_temp);
758         LL_delete_next(label_list);
759         memset(free_top_temp->label_name, 0, strlen(free_top_temp->label_name));
760         free(free_top_temp->label_name);
761         free_top_temp->label_name = NULL;
762         free(free_top_temp);
763         free_top_temp = NULL;
764         free_temp = NULL;
765     }
766     LL_free_head(label_list);
767 }
768
769 }
```