Main function:

```
int main(){
    pass1();
    pass2();
}
```

Pass1: 讀入 source.txt 並判斷該行的 location 的值為多少，寫入 location.txt 中，同時寫入 symbol_table.txt。

```
42          while(!feof(sFile)){
43              c = fgetc(sFile);           //to ensure whether the label value is null
44              if(c!='\t'){
45                  ungetc(c,sFile);        //put back the char just read
46                  fscanf(sFile, "%s", label);     //read label
47              }
48              else{
49                  memset(label, '\0', bufSize);   //the label is null
50              }
51              fscanf(sFile, "%s", directive);     //read directive
52              c = fgetc(sFile);           //to ensure whether the value is null
53              if(c!='\n'){
54                  ungetc(c,sFile);        //put back the char just read
55                  fscanf(sFile, "%s", buffer);
56                  c = fgetc(sFile);       //read '\n'
57              }
58              else{
59                  memset(buffer, '\0', bufSize);  //the value is null
60              }
61              if(start&&strcmp(directive,"END"))  //if the directive is END, fprintf to location.txt
62                  fprintf(locFile, "%x\t%s\t%s\t%s\n", loc, label, directive, buffer);
63              if(start&&label[0]!='\0')   //if the label is not null, fprintf to symbol_table.txt
64                  fprintf(symbolFile, "%s\t%x\n", label, loc);
65              if(!strcmp(directive,"START")){
66                  loc=strtol(buffer,NULL,16);     //change decimal to hexdecimal
67                  startLoc = loc;                 //store the first location value
68                  fprintf(locFile, "%x\t%s\t%s\t%s\n", loc, label, directive, buffer);
69                  start = true;
70              }
71              else if(!strcmp(directive,"BYTE")){
72                  int i = -1, record = 0;
73                  char type=buffer[0];
74                  while (buffer[++i] != '\'');
75                  while (buffer[++i] != '\'')
76                      record++;                   //store the length of the BYTE
77                  if(type=='X')
78                      record /= 2;
```

```
79                  loc += record;
80              }
81              else if(!strcmp(directive,"RESW")){
82                  int tmp = atoi(buffer);         //buffer stores the number of reserved words
83                  tmp *= 3;                       //one word equals to three bytes
84                  loc += tmp;
85              }
86              else if(!strcmp(directive,"RESB")){
87                  int tmp = atoi(buffer);         //buffer stores the number of reserved bytes
88                  loc += tmp;
89              }
90              else if(!strcmp(directive,"END")){
91                  lastLoc = loc;                  //store the last locaiton value
92                  fprintf(locFile, "\t%s\t%s\t%s", label, directive, buffer);
93                  break;
94              }
95              else{
96                  loc += 3;
97              }
98          }
99      }
```

Pass2: 先將 opcode.txt 和 symbol_table.txt 讀進陣列，再逐行走訪 locaiton.txt，判斷該行的 obcode 的值並寫入 source_program.txt 中，同時記錄要寫入 object_program.txt 的值，等到 object_program.txt 裡的 text record 達到換行的條件，寫入 text record，最後等到 directive 為 END 時，寫入 end record，程式結束。

```
133     while (true){
134         memset(obCode, '\0', 6);                //clear the value stored in obCode
135         c = fgetc(locFile);                     //check if the location value is null
136         if(c!='\t'){
137             ungetc(c,locFile);                  //put back the char just read
138             fscanf(locFile, "%s", loc);         //read the location value
139             c = fgetc(locFile);                 //read '\t'
140         }
141         else{
142             memset(loc, '\0', bufSize);         //the locaiton value is null
143         }
144         c = fgetc(locFile);                     //check if the label value is null
145         if(c!='\t'){
146             ungetc(c,locFile);                  //put back the char just read
147             fscanf(locFile, "%s", label);       //read the label
148         }
149         else{
150             memset(label, '\0', bufSize);       //the label value is null
151         }
152         fscanf(locFile, "%s", directive);       //read the directive
153         c = fgetc(locFile);                     //read the '\t'
154         c = fgetc(locFile);                     //check if the value is null
155         if(c!='\n'){
156             ungetc(c,locFile);                  //put back the char just read
157             fscanf(locFile, "%s", buffer);
158             c = fgetc(locFile);                 //read '\n'
159         }
160         else{
161             memset(buffer, '\0', bufSize);      //the value is null
162         }
163         if(!strcmp(directive, "BYTE")){
164             int j = -1;
165             if(buffer[0]=='C'){
166                 int i = 0;
167                 while (buffer[++j] != '\'');
168                 while (buffer[++j] != '\''){
169                     char tmp[2];
```

```
                    itoa((int)buffer[j], tmp, 16); //change (int)buffer[j] to hexdecimal and store in tmp
                    obCode[i++] = tmp[0];           //store the opCode in obCode
                    obCode[i++] = tmp[1];
                }
            }
            else if(buffer[0]=='X'){
                int tmp = 0;
                while (buffer[++j] != '\'');        //directly store the value in ' ' as obCode
                while (buffer[++j] != '\''){
                    obCode[tmp++]=buffer[j];
                }
            }
        }
        else if(!strcmp(directive, "WORD")){
            int obCodeTmp = 6;
            int bufferSize = (int)(strlen(buffer));
            char tmp[bufferSize];
            itoa(atoi(buffer),tmp , 16);
            for (i = bufferSize-1; i >= 0; i--){
                obCode[--obCodeTmp] = tmp[i];
            }
            for (i = obCodeTmp-1; i >= 0;i--)
                obCode[i] = '0';
        }
        else if(!strcmp(directive, "RESB")||!strcmp(directive, "RESW")||!strcmp(directive, "START")||!strcmp(directive, "END")){
            obCode[0] = '\0';
        }
        else{
            for (i = 0; i < opEnd;i++){
                if(!strcmp(directive,directiveBuf[i]))//find the correspond opCode
                    break;
            }
            strcat(obCode, opcodeBuf[i]);
            if (buffer[strlen(buffer) - 2] == ',' && buffer[strlen(buffer) - 1] == 'X'){//modify the obCode
                int tmpSize = (int)(strlen(buffer) - 2);
                char tmp[tmpSize];
                for (i = 0; i < tmpSize; i++){
```

```
                    tmp[i] = buffer[i];
                }
                for (i = 0; i < locEnd; i++){
                    if(!strcmp(tmp,labelBuf[i]))
                        break;
                }
                char str[6];
                itoa(locCounter[i]+4096*8, str, 16);        //plus 1 in x field in object code
                strcat(obCode, str);
            }
            else if(buffer[0]=='\0'){           //for the RSUB directive, complete the rest obCode as '0'
                for (i = 2; i < 6;i++)
                    obCode[i] = '0';
            }
            else{
                for (i = 0; i < locEnd;i++){
                    if(!strcmp(buffer,labelBuf[i]))
                        break;
                }
                char str[6];
                itoa(locCounter[i] , str, 16);  //find the correspond location
                strcat(obCode, str);
            }
        }
        for (i = 0; i < strlen(obCode);i++){
            if((int)(obCode[i])>=97&&(int)(obCode[i])<=122)     //change lower case to upper case
                obCode[i] -= 32;
        }
        if((int)(strlen(buffer))>=8)
            fprintf(sFile, "%s\t%s\t%s\t%s\t%s\n", loc,label,directive,buffer,obCode);
        else
            fprintf(sFile, "%s\t%s\t%s\t%s\t\t%s\n", loc,label,directive,buffer,obCode);
                        //write in object_program.txt
        if(!start){           //write header record
            start = true;
            fprintf(obFile, "%c%s\t%06X%06X\n", 'H', label, startLoc, (lastLoc - startLoc));
```

```
            memset(eachStartLoc, '0', 6);
            int tmp = (int)strlen(loc);
            for (i = tmp-1; i >=0 ;i--){
                eachStartLoc[2+i] = loc[i];
            }
        }
        else if(countLength+(strlen(obCode) / 2)>30||!strcmp(directive,"END")||!strcmp(directive,"RESW")||!strcmp(directive,"RESB")){
            //if the length of text record>69 or the directive is RESW or RESB, write text line.
            if(stringObProgram[0]!='\0'){                       //write text record
                fprintf(obFile, "%c%6s%02X%s\n", 'T', eachStartLoc, countLength, stringObProgram);
            }
            memset(eachStartLoc, '0', 6);
            for (i = 3; i >=0 ;i--){
                eachStartLoc[2+i] = loc[i];
            }
            if(!strcmp(directive,"RESW"))                       //represent the starting address should be modified
                startAddrModify = true;
            else if(!strcmp(directive,"RESB"))
                startAddrModify = true;
            countLength = strlen(obCode) / 2;
            memset(stringObProgram, '\0', 60);
            strcat(stringObProgram, obCode);
        }
        else{
            if(startAddrModify){                                //modify the starting address
                startAddrModify = false;
                memset(eachStartLoc, '0', 6);
                for (i = 3; i >=0 ;i--){
                    eachStartLoc[2+i] = loc[i];
                }
            }
            countLength += strlen(obCode)/2;                    //count the length
            strcat(stringObProgram, obCode);
        }
        if(!strcmp(directive,"END")){
            fprintf(obFile, "%c%06X", 'E', startLoc);
            break;
        }
```

Pass1:

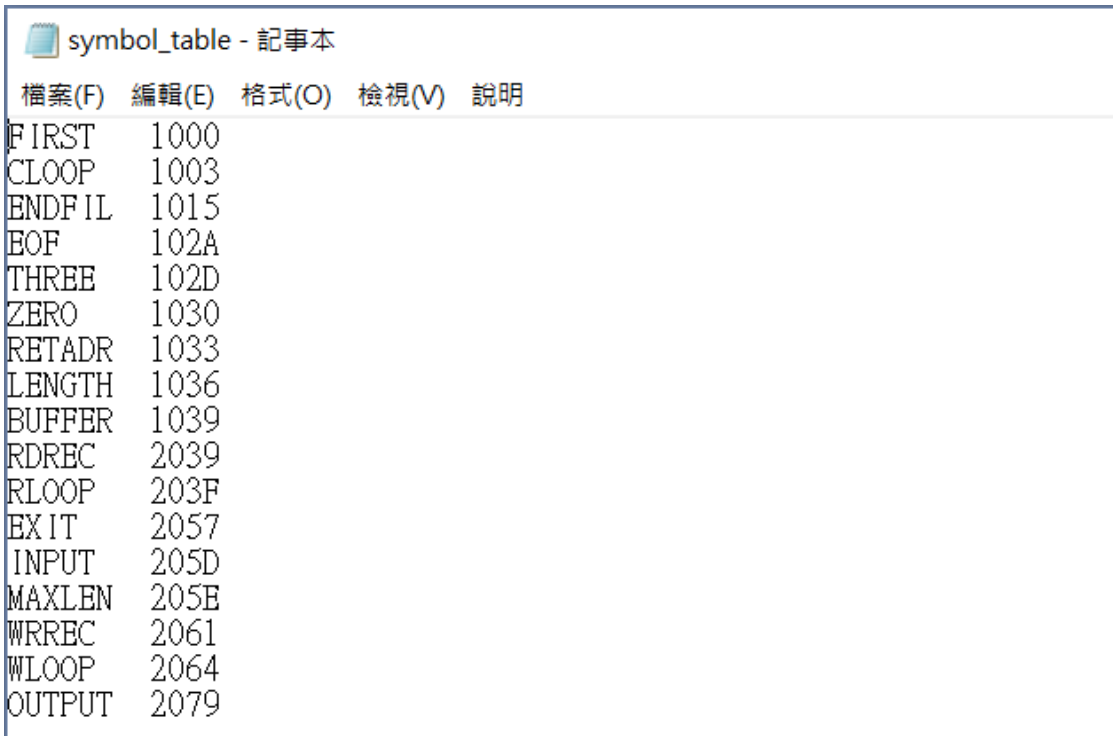
1. instruction 的 location counter 以及對應 source program

location - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
1000    COPY    START   1000
1000    FIRST   STL     RETADR
1003    CLOOP   JSUB    RDREC
1006            LDA     LENGTH
1009            COMP    ZERO
100C            JEQ     ENDFIL
100F            JSUB    WRREC
1012            J       CLOOP
1015    ENDFIL  LDA     EOF
1018            STA     BUFFER
101B            LDA     THREE
101E            STA     LENGTH
1021            JSUB    WRREC
1024            LDL     RETADR
1027            RSUB
102A    EOF     BYTE    C'EOF'
102D    THREE   WORD    3
1030    ZERO    WORD    0
1033    RETADR  RESW    1
1036    LENGTH  RESW    1
1039    BUFFER  RESB    4096
2039    RDREC   LDX     ZERO
203C            LDA     ZERO
203F    RLOOP   TD      INPUT
2042            JEQ     RLOOP
2045            RD      INPUT
2048            COMP    ZERO
204B            JEQ     EXIT
204E            STCH    BUFFER,X
2051            TIX     MAXLEN
2054            JLT     RLOOP
2057    EXIT    STX     LENGTH
205A            RSUB
205D    INPUT   BYTE    X'F1'
205E    MAXLEN  WORD    4096
2061    WRREC   LDX     ZERO
2064    WLOOP   TD      OUTPUT
2067            JEQ     WLOOP
206A            LDCH    BUFFER,X
206D            WD      OUTPUT
2070            TIX     LENGTH
2073            JLT     WLOOP
2076            RSUB
2079    OUTPUT  BYTE    X'05'
                END     FIRST
```

2. symbol table



Pass2:

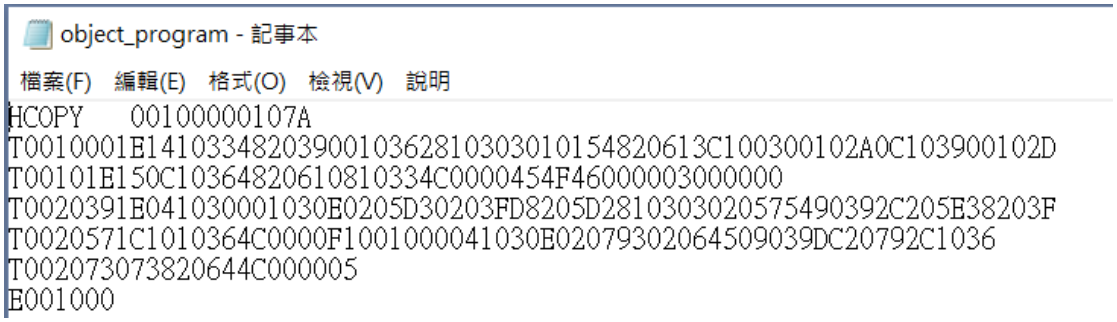
1. source program

source_program - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
1000    COPY    START   1000
1000    FIRST   STL     RETADR          141033
1003    CLOOP   JSUB    RDREC           482039
1006            LDA     LENGTH          001036
1009            COMP    ZERO            281030
100C            JEQ     ENDFIL          301015
100F            JSUB    WRREC           482061
1012            J       CLOOP           3C1003
1015    ENDFIL  LDA     EOF             00102A
1018            STA     BUFFER          0C1039
101B            LDA     THREE           00102D
101E            STA     LENGTH          0C1036
1021            JSUB    WRREC           482061
1024            LDL     RETADR          081033
1027            RSUB                    4C0000
102A    EOF     BYTE    C'EOF'          454F46
102D    THREE   WORD    3               000003
1030    ZERO    WORD    0               000000
1033    RETADR  RESW    1
1036    LENGTH  RESW    1
1039    BUFFER  RESB    4096
2039    RDREC   LDX     ZERO            041030
203C            LDA     ZERO            001030
203F    RLOOP   TD      INPUT           E0205D
2042            JEQ     RLOOP           30203F
2045            RD      INPUT           D8205D
2048            COMP    ZERO            281030
204B            JEQ     EXIT            302057
204E            STCH    BUFFER,X        549039
2051            TIX     MAXLEN          2C205E
2054            JLT     RLOOP           38203F
2057    EXIT    STX     LENGTH          101036
205A            RSUB                    4C0000
205D    INPUT   BYTE    X'F1'           F1
205E    MAXLEN  WORD    4096            001000
2061    WRREC   LDX     ZERO            041030
2064    WLOOP   TD      OUTPUT          E02079
2067            JEQ     WLOOP           302064
206A            LDCH    BUFFER,X        509039
206D            WD      OUTPUT          DC2079
2070            TIX     LENGTH          2C1036
2073            JLT     WLOOP           382064
2076            RSUB                    4C0000
2079    OUTPUT  BYTE    X'05'           05
                END     FIRST
```

2. object program


object_program - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
HCOPY    00100000107A
T0010001E141033482039001036281030301015482061 3C100300102A0C103900102D
T00101E150C10364820610810334C0000454F46000003000000
T0020391E041030001030E0205D30203FD8205D2810303020575490392C205E38203F
T0020571C1010364C0000F1001000041030E02079302064509039DC20792C1036
T002073073820644C000005
E001000
```