第一次作業_lex 結報

資訊三丁 徐葆驊 D1053020

1.作法

- Int 規則:
  - [-]?[0-9]+[;]?
    - 先判斷有沒有正負號開頭，中間是判斷有沒有多個數字，最後判斷有沒有接分號。
  - [-]?[0-9]+/[\n]
    - 跟前面的設計一樣，但是最後是要接換行符號。
- Float 規則:
  - [-]?[0-9]+[.][0-9]+[;]?
    - 先判斷有沒有正負號開頭，中間是判斷有沒有多個數字加一個小數點再來有沒有多個數字，最後判斷有沒有接分號。
  - [-]?[0-9]+[.][0-9]+/[\n]
    - 跟前面的設計一樣，但是最後是要接換行符號。
- ID 規則:
  - [a-zA-Z][a-zA-Z0-9_]*[;]?
    - 先一次字母開頭，再來接多次字母或數字或底線，最後判斷有沒有接分號。
  - [a-zA-Z][a-zA-Z0-9_]*/[\n]

        - 跟前面的設計一樣，但是最後是要接換行符號。

- Error 規則:
    - [_][^;\n]*[;]?
        - 底線當頭的時候
    - [_][^;\n]*/\n
        - 底線當頭，最後換行
    - [0-9]+([a-zA-Z0-9]|[_])[^;\n]*[;]?
        - 數字當頭，接底線或字母
    - [0-9]+([a-zA-Z0-9]|[_])[^;\n]*/\n
        - 數字當頭，接底線或字母，最後換行
    - [^;\n]*[.][^;\n]*[;] ?
        - 點當頭或是不是數字的當頭
    - [^;\n]*[.][^;\n]*/\n
        - 點當頭或是不是數字的當頭，最後換行
    - [^;\n]*[.][^;\n]*[.][;] ?
        - 有兩個點以上
    - [^;\n]*[.][^;\n]*[.]/\n
        - 有兩個點以上，最後換行
    - [-][^;\n]*[;]?
        - 減號當頭
    - [-][^;\n]*/\n
        - 減號當頭，最後換行
    - [^;\n]*[-][^;\n]*[;]
        - 減號在其中
    - [^;\n]*[-][^;\n]*/\n
        - 減號在其中，最後換行
    - [;]
        - 純分號

2.程式碼

基本題程式碼:

```
%{
#include <stdio.h>
int Total_Num=0;
%}
%option noyywrap
%%
[-]?[0-9]+[;]? {
    printf("Integer: %s\n",yytext);
}
[-]?[0-9]+/[\n] {
    printf("Integer: %s\n",yytext);
}

[-]?[0-9]+[.][0-9]+[;]? {
    printf("Float: %s\n",yytext);
```

```
16. }
17. [-]?[0-9]+[.][0-9]+/[\n] {
18.     printf("Float: %s\n",yytext);
19. }
20.
21. [a-zA-Z][a-zA-Z0-9_]*[;]? {
22.     printf("ID: %s\n",yytext);
23. }
24. [a-zA-Z][a-zA-Z0-9_]*/[\n] {
25.     printf("ID: %s \n",yytext);
26. }
27.
28. [_][^;\n]*[;]? {
29.     printf("Error: %s \n",yytext);   //底線當頭
30. }
31. [_][^;\n]*/\n {
32.     printf("Error: %s \n",yytext);   //底線當頭，最後換行
33. }
34.
35. [0-9]+([a-zA-Z0-9]|[_])[^;\n]*[;]? {
36.     printf("Error: %s \n",yytext);   //數字當頭，接底線或字母
```

```
37. }
38. [0-9]+([a-zA-Z0-9]|[_])[^;\n]*/\n {
39.     printf("Error: %s \n",yytext);  //數字當頭，接底線或字母，最後換行
40. }
41.
42. [^;\n]*[.][^;\n]*[;]? {
43.     printf("Error: %s \n",yytext); //點當頭
44. }
45. [^;\n]*[.][^;\n]*/\n {
46.     printf("Error: %s \n",yytext); //點當頭，最後換行
47. }
48.
49. [^;\n]*[.][^;\n]*[.][;]? {
50.     printf("Error: %s \n",yytext); //有兩個點以上
51. }
52. [^;\n]*[.][^;\n]*[.]/\n {
53.     printf("Error: %s \n",yytext); //有兩個點以上，最後換行
54. }
55.
56. [-][^;\n]*[;]? {
57.     printf("Error: %s \n",yytext); //減當頭
```

```
58. }
59. [-][^;\n]*/\n {
60.     printf("Error: %s \n",yytext); //減當頭，最後換行
61. }
62.
63. [^;\n]*[-][^;\n]*[;]? {
64.     printf("Error: %s \n",yytext); //減在其中
65. }
66. [^;\n]*[-][^;\n]*/\n {
67.     printf("Error: %s \n",yytext); //減在其中，最後換行
68. }
69.
70. [;] {
71.     printf("Error: %s \n",yytext); //純分號
72. }
73. \n {}
74. %%
75. int main()
76. {
77.     yylex();
```

```
78.     return 0;
79. }
```

加分題程式碼:

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  #include <string.h>
4.  #define SIZE 1000
5.
6.  int totalInt = 0;
7.  int totalFloat = 0;
8.  int totalID = 0;
9.  int totalError = 0;
10.
11. int IsInt(char token[]){
12.     for(int i = 0; i < strlen(token); i++){
13.         switch (token[i])
14.         {
15.         case '0':
```

```
16.                 break;
17.             case '1':
18.                 break;
19.             case '2':
20.                 break;
21.             case '3':
22.                 break;
23.             case '4':
24.                 break;
25.             case '5':
26.                 break;
27.             case '6':
28.                 break;
29.             case '7':
30.                 break;
31.             case '8':
32.                 break;
33.             case '9':
34.                 break;
```

```
35.             case '-'://負號
36.                 if(i != 0 ){return 0;} //如果不是第一個字元有負號，則 error
37.                 break;
38.             case ';'://分號
39.                 //避免第一個字元就是分號
40.                 if(i == 0){return 0;}
41.                 break;
42.             case '\n'://換行
43.                 //避免第一個字元就是換行
44.                 if(i == 0){return 0;}
45.                 break;
46.             default://除了數字和負號以外的字元都是 error
47.                 return 0;
48.                 break;
49.             }
50.     }
51.     return 1;
52. }
53. int IsFloat(char token[]){
```

```
54.     int dotCount = 0;//小數點個數
55.     for(int i = 0; i < strlen(token); i++){
56.         switch (token[i])
57.         {
58.         case '0':
59.             break;
60.         case '1':
61.             break;
62.         case '2':
63.             break;
64.         case '3':
65.             break;
66.         case '4':
67.             break;
68.         case '5':
69.             break;
70.         case '6':
71.             break;
72.         case '7':
```

```
73.                 break;
74.             case '8':
75.                 break;
76.             case '9':
77.                 break;
78.             case '.':
79.                 dotCount++;
80.                 if(dotCount > 1){return 0;}
81.                 break;
82.             case '-'://負號
83.                 if(i != 0 ){return 0;} //如果不是第一個字元有負號，則 error
84.                 break;
85.             case '_'://底線
86.                 return 0;
87.                 break;
88.             case ';'://分號
89.                 //避免第一個字元就是分號
90.                 if(i == 0){return 0;}
91.                 break;
```

```
92.             case '\n'://換行
93.                 //避免第一個字元就是換行
94.                 if(i == 0){return 0;}
95.                 break;
96.             default:
97.                 return 0;
98.                 break;
99.           }
100.      }
101.      return 1;
102.
103.  }
104.  int IsID(char token[]){
105.      //不符合規則的開頭，負號、底線、小數點或數字開頭
106.      switch (token[0])
107.      {
108.      case '-'://負號
109.          return 0;
110.          break;
111.      case '_'://底線
```

```
112.             return 0;
113.             break;
114.         case '.'://小數點
115.             return 0;
116.             break;
117.         case ';'://分號，避免第一個字元就是分號
118.             return 0;
119.             break;
120.         case '\n'://換行
121.             //避免第一個字元就是換行
122.             return 0;
123.             break;
124.         case '0':
125.             return 0;
126.             break;
127.         case '1':
128.             return 0;
129.             break;
130.         case '2':
```

```
131.            return 0;
132.            break;
133.        case '3':
134.            return 0;
135.            break;
136.        case '4':
137.            return 0;
138.            break;
139.        case '5':
140.            return 0;
141.            break;
142.        case '6':
143.            return 0;
144.            break;
145.        case '7':
146.            return 0;
147.            break;
148.        case '8':
149.            return 0;
```

```
150.            break;
151.        case '9':
152.            return 0;
153.            break;
154.        default:
155.            break;
156.        }
157.        //token body
158.        for(int i = 1; i < strlen(token); i++){
159.            switch (token[i])
160.            {
161.            case '-'://負號
162.                return 0;
163.                break;
164.            case '.'://小數點
165.                return 0;
166.                break;
167.            default:
168.                break;
```

```
169.            }
170.        }
171.        return 1;
172.    }
173.    void judgeToken(char token[]) {
174.        int isInt = 0;//是否為整數
175.        int isFloat = 0;//是否為浮點數
176.        int isID = 0;//是否為 ID
177.
178.        isInt = IsInt(token);
179.        isFloat = IsFloat(token);
180.        isID = IsID(token);
181.        if(isInt){
182.            totalInt++;
183.            if(token[strlen(token)-1] == '\n'){printf("Integer: %s", token);}
184.            else{printf("Integer: %s\n", token);}
185.        }else if(isFloat){
186.            totalFloat++;
187.            if(token[strlen(token)-1] == '\n')printf("Float: %s", token);
188.            else{printf("Float: %s\n", token);}
```

```
189.        }else if(isID){
190.            totalID++;
191.            if(token[strlen(token)-1] == '\n')printf("ID: %s", token);
192.            else{printf("ID: %s\n", token);}
193.        }else{
194.            totalError++;
195.            if(token[strlen(token)-1] == '\n')printf("Error: %s", token);
196.            else{printf("Error: %s\n", token);}
197.        }
198.
199.    }
200.
201.    int main() {
202.        FILE *fp;
203.        char token[SIZE];
204.        memset(token, '\0', strlen(token));
205.        char input[2];
206.        input[1] = '\0';
207.        int printFlag = 0;
208.        if (( fp = fopen("testText.txt", "r")) == NULL)
```

```
209.    {
210.        printf("Error! opening file");
211.        // 文件指针返回 NULL 则退出
212.        exit(1);
213.    }
214.    input[0] = fgetc(fp);
215.    while (input[0] != EOF)
216.    {
217.        strcat(token, input);
218.        if(input[0] == ';'){
219.            judgeToken(token);
220.            memset(token, '\0', strlen(token));
221.            printFlag = 1;
222.        }else if(input[0] == '\n'){
223.            if(printFlag == 0){
224.                judgeToken(token);
225.            }else{
226.                printFlag = 0;
227.            }
```

```
228.                 memset(token, '\0', strlen(token));
229.             }else{//印完分號下一個不是換行就重置 flag
230.                 printFlag = 0;
231.             }
232.             input[0] = fgetc(fp);
233.         }
234.         //最後一個 token
235.         judgeToken(token);
236.         fclose(fp);
237.         for(int i = 0; i < 50; i++){
238.             printf("*");
239.         }
240.         printf("\n");
241.         printf("Total Integer: %d\n", totalInt);
242.         printf("Total Float: %d\n", totalFloat);
243.         printf("Total ID: %d\n", totalID);
244.         printf("Total Error: %d\n", totalError);
245.         return 0;
246.     }
```

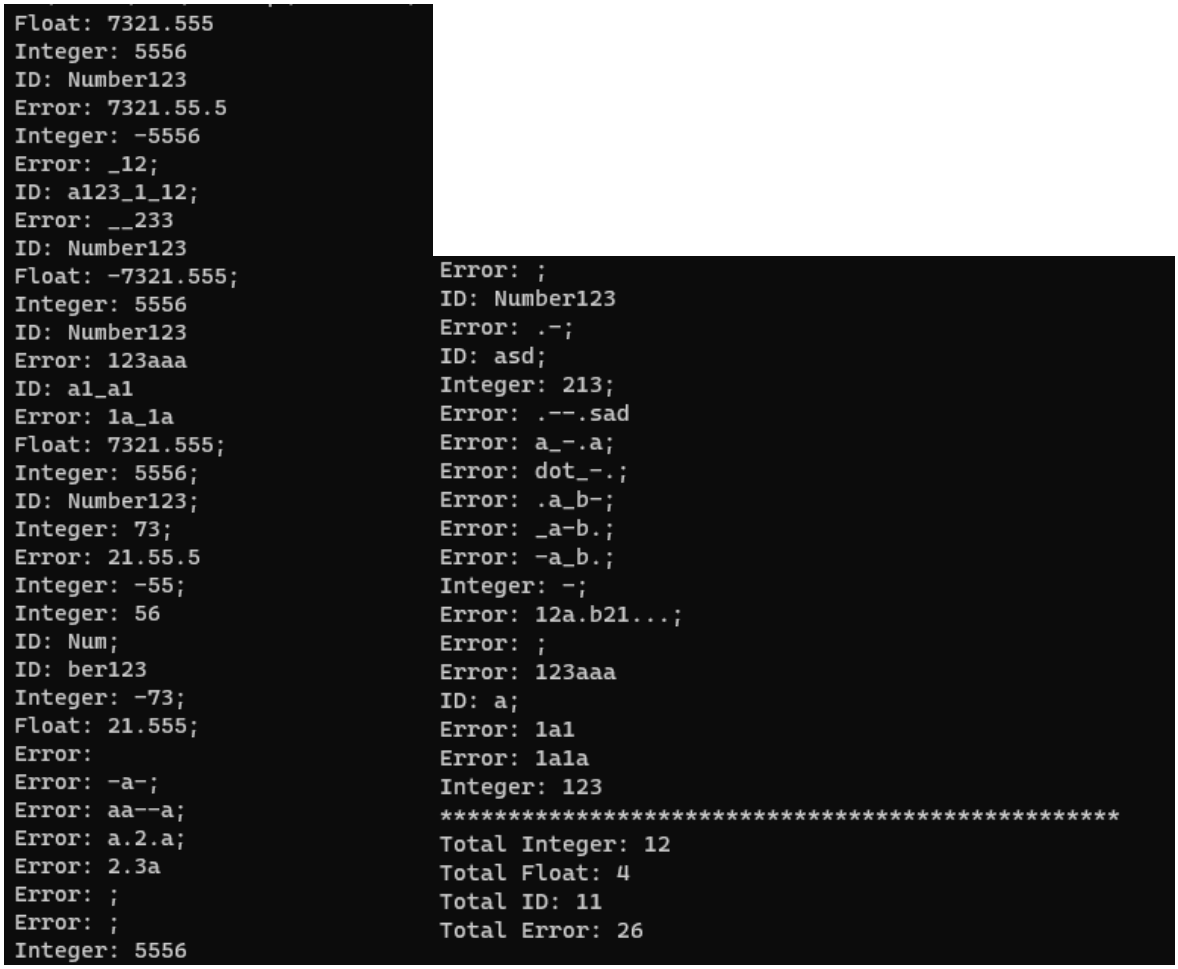
3.執行結果

基本題執行結果:

```
Float: 7321.555
Integer: 5556
ID: Number123
Error: 7321.55.5
Integer: -5556
Error: _12;
ID: a123_1_12;
Error: __233
ID: Number123
Float: -7321.555;
Integer: 5556
ID: Number123
Error: 123aaa
ID: a1_a1
Error: 1a_1a
Float: 7321.555;
Integer: 5556;
ID: Number123;
Integer: 73;
Error: 21.55.5
Integer: -55;
Integer: 56
ID: Num;
ID: ber123
Integer: -73;
Float: 21.555;
Error: -a-;
Error: aa--a;
Error: a.2.a;
Error: 2.3a
Error: ;
Error: ;
Integer: 5556
Error: ;
ID: Number123
Error: .-;
ID: asd;
```

```
Integer: 213;
Error: .--.sad
Error: a_-.a;
Error: dot_-.;
Error: .a_b-;
Error: _a-b.;
Error: -a_b.;
Error: -;
Error: 12a.b21...;
Error: ;
Error: 123aaa
ID: a;
Error: 1a1
Error: 1a1a
Integer: 123
```

加分題的執行結果:


Float: 7321.555
Integer: 5556
ID: Number123
Error: 7321.55.5
Integer: -5556
Error: _12;
ID: a123_1_12;
Error: __233
ID: Number123
Float: -7321.555;
Integer: 5556
ID: Number123
Error: 123aaa
ID: a1_a1
Error: 1a_1a
Float: 7321.555;
Integer: 5556;
ID: Number123;
Integer: 73;
Error: 21.55.5
Integer: -55;
Integer: 56
ID: Num;
ID: ber123
Integer: -73;
Float: 21.555;
Error:
Error: -a-;
Error: aa--a;
Error: a.2.a;
Error: 2.3a
Error: ;
Error: ;
Integer: 5556
Error: ;
ID: Number123
Error: .-;
ID: asd;
Integer: 213;
Error: .--.sad
Error: a_-.a;
Error: dot_-.;
Error: .a_b-;
Error: _a-b.;
Error: -a_b.;
Integer: -;
Error: 12a.b21...;
Error: ;
Error: 123aaa
ID: a;
Error: 1a1
Error: 1a1a
Integer: 123
****
Total Integer: 12
Total Float: 4
Total ID: 11
Total Error: 26

4.討論&心得:

- 遇到的困難:

我覺得主要是在判斷 error 的情況，本來想說可以將除了 int、float、ID 以外的情況都列為 error，但是發現好像沒有這種規則。後來想說可以先辨識完符合規則的字然後其他的字".*”都當 error，但是 yytext 很像 multicharacter lookahead，會取得最長的符合規則的字串，所以這個做法也不行，所以最後選擇窮舉出全部的可能。

後來我發現我的演算法，因為後面一定要接分號或是換行符號才會判斷為一個 token，所以讀到測資的最後如果沒有分號而是 EOF，就會出問題，但是後來把每一次判斷有沒有分號後面接?，就可以判斷 0 或 1 次就解決這個問題了。

一開始的時候想說可以用|來判斷一個 token 是接分號還是換行，但是發現用|的規則好像怪怪的，後來就直接把牠們分開寫了。