

# 编译原理第一次实验测试用例：目录

<b>1</b>	<b>A 组测试用例</b>	<b>3</b>
1.1	A-1 . . . . .	3
1.2	A-2 . . . . .	3
1.3	A-3 . . . . .	4
1.4	A-4 . . . . .	4
1.5	A-5 . . . . .	4
1.6	A-6 . . . . .	5
1.7	A-7 . . . . .	5
1.8	A-8 . . . . .	6
1.9	A-9 . . . . .	7
1.10	A-10 . . . . .	8
<b>2</b>	<b>B 组测试用例</b>	<b>9</b>
2.1	B-1 . . . . .	9
2.2	B-2 . . . . .	10
<b>3</b>	<b>C 组测试用例</b>	<b>12</b>
3.1	C-1 . . . . .	12
3.2	C-2 . . . . .	19
<b>4</b>	<b>D 组测试用例</b>	<b>39</b>
4.1	D-1 . . . . .	39
4.2	D-2 . . . . .	43
4.3	D-3 . . . . .	46
<b>5</b>	<b>E 组测试用例</b>	<b>50</b>
5.1	E1-1 . . . . .	50
5.2	E1-2 . . . . .	51
5.3	E2-1 . . . . .	57
5.4	E2-2 . . . . .	58
5.5	E3-1 . . . . .	60
5.6	E3-2 . . . . .	62



## 1 A 组测试用例

本组测试用例共 10 个，每个仅包含单个的词法或者语法错误。除特殊说明外，不可多报。多报、漏报错误，或者打印语法树都会导致扣分。错误编号和行号之后的说明文字不要求与给出的输出完全一致，仅供助教理解使用，不作为评分依据。

### 1.1 A-1

输入

```
1  int main () {  
2      int a = 1;  
3      int b = a@2;  
4  }
```

输出

```
1 Error type A at line 3: Mysterious character '@'.
```

说明：也可以报成 B 类错误。

### 1.2 A-2

输入

```
1  int foo(int x) {  
2      return x + 1;  
3  }  
4  
5  int main() {  
6      int 3c = foo(1);  
7  }
```

输出

```
1 Error type A at line 6: Illegal identifier '3c'.
```

说明：也可以报成 B 类错误。

### 1.3 A-3

输入

```
1 struct Person {  
2     int x;  
3     int y;  
4 };  
5  
6 int main() {  
7     Person person;  
8     int a = person.x + 1;  
9 }
```

输出

```
1 Error type B at line 7: Illegal type specifier.
```

说明：缺少 `struct` 关键字。

### 1.4 A-4

输入

```
1 int a[10][10];  
2 int b = 100;  
3  
4 int main(){  
5     int c = b + a[0][0];  
6 }
```

输出

```
1 Error type B at line 2: Global variable cannot have initializer.
```

说明：全局变量定义时不能初始化。

### 1.5 A-5

输入

```

1  int main(){
2      float f = 1.2.2.4;
3  }

```

输出

```

1  Error type A at line 2: Illegal number '1.2.2.4'.

```

说明：也可以报成 B 类错误。

## 1.6 A-6

输入

```

1  struct Taxi {
2      int id;
3      float speed;
4      float location;
5  } taxis[10];
6
7  int main(){
8      float speed = taxis[0,1].speed;
9  }

```

输出

```

1  Error type B at line 8: Array index can only be an integer.

```

## 1.7 A-7

输入

```

1  int array[10];
2
3  int initArray(int len){
4      int i = 0;
5      while(i < len){
6          array[i] = i;

```

```

7         i = i + 1;
8     }
9 }
10
11 int swapArray(int len,){
12     int i = 0;
13     while(i < len / 2){
14         int temp = array[i];
15         array[i] = array[len - i - 1];
16         array[len - i - 1] = temp;
17     }
18 }
19
20 int main(){
21     initArray(10);
22     bubble_sort(10);
23 }

```

输出

```

1 Error type B at line 11: Expect more parameters after ', '.

```

## 1.8 A-8

输入

```

1 struct Oops say() {
2     struct Point {
3         int x;
4         int y;
5         int z;
6     } p1, p2;
7     return 0;
8 }
9

```

```

10  int;
11  int;
12
13  struct st {
14      int s1;
15      float s2;
16      struct st s3, s4;
17      int arr[10];
18  };
19
20  int main(){
21      struct st st1;
22      int i = 0;
23      while(i < 10){
24          st1.arr[i] *= 100;
25      }
26  }

```

输出

```

1 Error type B at line 24: ' *= ' is not supported.

```

说明：c-不支持'\*= ' 运算符

## 1.9 A-9

输入

```

1  int array[10];
2  int a,b,d,c;
3
4  struct Two{
5      int x, y;
6  };
7
8  int compare(struct Two t1, struct Two t2){

```

```

9      if(t1.x > t2.x && t1.y > t2.y)  else return 0;
10  }
11
12  int main(){
13      struct Two t1,t2;
14      compare(t1, t2);
15  }

```

输出

```

1  Error type B at line 9: Expect statement between 'if' and 'else'.

```

## 1.10 A-10

输入

```

1  int a,b;
2
3  int add(int x, int y){
4      return x + y;
5  }
6
7  int minus(int x, int y){
8      return x - y;
9  }
10
11 int cal(int x, int y){
12     return;
13 }
14
15 int main(){
16     a = 1;
17     b = 2;
18     cal(add(a,b), minus(a,b));
19 }

```



输出

```
1 Error type B at line 12: Expect an Exp between return and ;
```

说明：return 关键字后需要有 Exp。

## 2 B 组测试用例

本组测试用例共 2 个，每个用例包含多处不同的错误。除特殊说明外，漏报、多报错误或者打印语法树都会导致扣分。

### 2.1 B-1

输入

```
1 int foo(int arr[10], int y, struct player p){
2     int a,b,c;
3     struct st{
4         int _i, _j, _k;
5         float 2g;
6         int _l;
7     };
8     int an = 100;
9     struct st t1;
10
11     while(a <= 100);
12
13     if(a > c){
14         return t2._i + t2._j;
15     }
16     else{
17         b = t1._i + an;
18     }
19
20     return 10;
21 }
```

```

22
23 int bar(float h1, float h2, float h3){
24     int list[100.0];
25     struct st{
26         float hhh;
27         float ttt;
28         float ppp;
29     } t;
30
31     list[1] = t.hhh;
32 }

```

输出

```

1 Error type B at Line 5: Illegal identifier '2g'.
2 Error type B at Line 7: Expect DecList after '}'.
3 Error type B at Line 11: Expect statement after 'while' condition.
4 Error type B at Line 24: Illegal array index '100.0'.

```

说明：第 5 行 **identifier** 不能以数字开头；第 7 行在函数中不能单纯声明结构体；第 11 行不支持 **while** 循环体为空；第 24 行数组索引只能为整数。

## 2.2 B-2

输入

```

1 int shsort(int s[4], int n)
2 {
3     int i, j, d;
4     d = n / 2;
5     while(d >= 1)
6     {
7         i = d + 1;
8         while (i <= n)
9         {
10             s[0] = s[i];

```

```

11         j = i - d;
12         while((j > 0) && (s[0] < s[j]))
13         {
14             s[j + d] = s[j];
15             j = j - d;
16         }
17         s[j + d] = s[0];
18         i++;
19     }
20     d = d / 2;
21 }
22 return 0;
23 }
24 int main()
25 {
26     int a[4];
27     a[0] = 2;
28     a[1] = 4;
29     a[2] = 3;
30     a[3]] = 1;
31     a[3] = 10;
32     int n = 4;
33     shsort(a, n);
34     return 0;
35 }

```

### 输出

```

1 Error type B at Line 5: Unexpected ' ' between '>' and '='.
2 Error type B at Line 18: Unexpected '+' after '+'.
3 Error type B at Line 30: Unexpected ']' after ']''.
4 Error type B at Line 32: Unexpected 'int' specifier.

```

说明：第 5 行多了一个空格；第 18 行 '+' 后缺少 Exp；第 30 行多了一个 ']'，第 32 行声明只

能在开头。

### 3 C 组测试用例

本组测试用例共 2 个，不包含任何错误，需要输出正确的语法树。除特殊说明外，应与给出的语法树完全相同。语法树打印错误酌情扣分。

#### 3.1 C-1

输入

```
1  int MatrixMax(int a[3][4])
2  {
3      int i=0,j=0;
4      int max,max_i=0,max_j=0;
5      max = a[0][0];
6      while(i < 3)
7      {
8          while(j < 4)
9          {
10             if(a[i][j] > max)
11             {
12                 max=a[i][j];
13                 max_i=i;
14                 max_j=j;
15             }
16             j = j + 1;
17         }
18         i = i + 1;
19     }
20     return 0;
21 }
```

输出

```
1 Program (1)
```

```

2 ExtDefList (1)
3   ExtDef (1)
4     Specifier (1)
5       TYPE: int
6     FunDec (1)
7       ID: MatrixMax
8       LP
9       VarList (1)
10        ParamDec (1)
11          Specifier (1)
12            TYPE: int
13          VarDec (1)
14            VarDec (1)
15              VarDec (1)
16                ID: a
17                LB
18                INT: 3
19                RB
20                LB
21                INT: 4
22                RB
23          RP
24    CompSt (2)
25      LC
26      DefList (3)
27        Def (3)
28          Specifier (3)
29            TYPE: int
30          DecList (3)
31            Dec (3)
32              VarDec (3)
33                ID: i

```

```

34         ASSIGNOP
35         Exp (3)
36         INT: 0
37     COMMA
38     DecList (3)
39         Dec (3)
40         VarDec (3)
41         ID: j
42         ASSIGNOP
43         Exp (3)
44         INT: 0
45     SEMI
46     DefList (4)
47     Def (4)
48         Specifier (4)
49         TYPE: int
50         DecList (4)
51         Dec (4)
52         VarDec (4)
53         ID: max
54     COMMA
55     DecList (4)
56         Dec (4)
57         VarDec (4)
58         ID: max_i
59         ASSIGNOP
60         Exp (4)
61         INT: 0
62     COMMA
63     DecList (4)
64         Dec (4)
65         VarDec (4)

```

```

66             ID: max_j
67             ASSIGNOP
68             Exp (4)
69             INT: 0
70         SEMI
71 StmtList (5)
72     Stmt (5)
73     Exp (5)
74     Exp (5)
75         ID: max
76     ASSIGNOP
77     Exp (5)
78     Exp (5)
79     Exp (5)
80         ID: a
81     LB
82     Exp (5)
83     INT: 0
84     RB
85     LB
86     Exp (5)
87     INT: 0
88     RB
89 SEMI
90 StmtList (6)
91     Stmt (6)
92     WHILE
93     LP
94     Exp (6)
95     Exp (6)
96         ID: i
97     RELOP

```

```

98         Exp (6)
99         INT: 3
100    RP
101    Stmt (7)
102        CompSt (7)
103            LC
104            StmtList (8)
105                Stmt (8)
106                    WHILE
107                        LP
108                            Exp (8)
109                                Exp (8)
110                                    ID: j
111                                RELOP
112                                    Exp (8)
113                                        INT: 4
114                            RP
115                        Stmt (9)
116                            CompSt (9)
117                                LC
118                                    StmtList (10)
119                                        Stmt (10)
120                                            IF
121                                                LP
122                                                    Exp (10)
123                                                        Exp (10)
124                                                            Exp (10)
125                                                                Exp (10)
126                                                                    ID: a
127                                                                LB
128                                                                    Exp (10)
129                                                                        ID: i

```



130	RB
131	LB
132	Exp (10)
133	ID: j
134	RB
135	RELOP
136	Exp (10)
137	ID: max
138	RP
139	Stmt (11)
140	CompSt (11)
141	LC
142	StmtList (12)
143	Stmt (12)
144	Exp (12)
145	Exp (12)
146	ID: max
147	ASSIGNOP
148	Exp (12)
149	Exp (12)
150	Exp (12)
151	ID: a
152	LB
153	Exp (12)
154	ID: i
155	RB
156	LB
157	Exp (12)
158	ID: j
159	RB
160	SEMI
161	StmtList (13)

162	Stmt (13)
163	Exp (13)
164	Exp (13)
165	ID: max_i
166	ASSIGNOP
167	Exp (13)
168	ID: i
169	SEMI
170	StmtList (14)
171	Stmt (14)
172	Exp (14)
173	Exp (14)
174	ID: max_j
175	ASSIGNOP
176	Exp (14)
177	ID: j
178	SEMI
179	RC
180	StmtList (16)
181	Stmt (16)
182	Exp (16)
183	Exp (16)
184	ID: j
185	ASSIGNOP
186	Exp (16)
187	Exp (16)
188	ID: j
189	PLUS
190	Exp (16)
191	INT: 1
192	SEMI
193	RC

```

194         StmtList (18)
195         Stmt (18)
196         Exp (18)
197         Exp (18)
198         ID: i
199         ASSIGNOP
200         Exp (18)
201         Exp (18)
202         ID: i
203         PLUS
204         Exp (18)
205         INT: 1
206         SEMI
207     RC
208 StmtList (20)
209 Stmt (20)
210 RETURN
211 Exp (20)
212 INT: 0
213 SEMI
214 RC

```

说明：使用的空格可以用 Tab 替换，注意缩进

## 3.2 C-2

输入

```

1  int MergeSort(int arr[100], int tmp[100], int start, int end)
2  {
3      int mid;
4      if (start < end)
5      {
6          mid = (start + end) / 2;
7          MergeSort(arr, tmp, start, mid);

```

```

8         MergeSort(arr, tmp, mid + 1, end);
9         Merge(arr, tmp, start, mid, end);
10    }
11 }
12
13 int Merge(int arr[100], int tmp[100], int start, int mid, int end)
14 {
15     int i = start, j = mid + 1, k = start;
16     while (i != mid + 1 && j != end + 1)
17     {
18         if (arr[i] >= arr[j]){
19             tmp[k] = arr[j];
20             k = k + 1;
21             j = j + 1;
22         }
23         else{
24             tmp[k] = arr[i];
25             k = k + 1;
26             i = i + 1;
27         }
28     }
29     while (i != mid + 1){
30         tmp[k] = arr[i];
31         k = k + 1;
32         i = i + 1;
33     }
34     while (j != end + 1){
35         tmp[k] = arr[j];
36         k = k + 1;
37         j = j + 1;
38     }
39     i = start;

```

```

40     while(i <= end){
41         arr[i] = tmp[i];
42         i = i + 1;
43     }
44 }

```

## 输出

```

1 Program (1)
2 ExtDefList (1)
3   ExtDef (1)
4     Specifier (1)
5       TYPE: int
6     FunDec (1)
7       ID: MergeSort
8       LP
9       VarList (1)
10        ParamDec (1)
11          Specifier (1)
12            TYPE: int
13          VarDec (1)
14            VarDec (1)
15              ID: arr
16              LB
17              INT: 100
18              RB
19          COMMA
20          VarList (1)
21            ParamDec (1)
22              Specifier (1)
23                TYPE: int
24              VarDec (1)
25                VarDec (1)
26                  ID: tmp

```

```

27         LB
28         INT: 100
29         RB
30     COMMA
31     VarList (1)
32         ParamDec (1)
33             Specifier (1)
34                 TYPE: int
35             VarDec (1)
36                 ID: start
37     COMMA
38     VarList (1)
39         ParamDec (1)
40             Specifier (1)
41                 TYPE: int
42             VarDec (1)
43                 ID: end
44     RP
45 CompSt (2)
46     LC
47     DefList (3)
48         Def (3)
49             Specifier (3)
50                 TYPE: int
51             DecList (3)
52                 Dec (3)
53                     VarDec (3)
54                         ID: mid
55     SEMI
56 StmtList (4)
57     Stmt (4)
58     IF

```

```

59      LP
60      Exp (4)
61          Exp (4)
62              ID: start
63          RELOP
64      Exp (4)
65          ID: end
66      RP
67      Stmt (5)
68          CompSt (5)
69              LC
70              StmtList (6)
71                  Stmt (6)
72                      Exp (6)
73                          Exp (6)
74                              ID: mid
75                          ASSIGNOP
76                      Exp (6)
77                          Exp (6)
78                              LP
79                                  Exp (6)
80                                      Exp (6)
81                                          ID: start
82                                              PLUS
83                                                  Exp (6)
84                                                      ID: end
85                                      RP
86                                  DIV
87                                      Exp (6)
88                                          INT: 2
89                                  SEMI
90                          StmtList (7)

```

91	Stmt (7)
92	Exp (7)
93	ID: MergeSort
94	LP
95	Args (7)
96	Exp (7)
97	ID: arr
98	COMMA
99	Args (7)
100	Exp (7)
101	ID: tmp
102	COMMA
103	Args (7)
104	Exp (7)
105	ID: start
106	COMMA
107	Args (7)
108	Exp (7)
109	ID: mid
110	RP
111	SEMI
112	StmtList (8)
113	Stmt (8)
114	Exp (8)
115	ID: MergeSort
116	LP
117	Args (8)
118	Exp (8)
119	ID: arr
120	COMMA
121	Args (8)
122	Exp (8)



123	ID: tmp
124	COMMA
125	Args (8)
126	Exp (8)
127	Exp (8)
128	ID: mid
129	PLUS
130	Exp (8)
131	INT: 1
132	COMMA
133	Args (8)
134	Exp (8)
135	ID: end
136	RP
137	SEMI
138	StmtList (9)
139	Stmt (9)
140	Exp (9)
141	ID: Merge
142	LP
143	Args (9)
144	Exp (9)
145	ID: arr
146	COMMA
147	Args (9)
148	Exp (9)
149	ID: tmp
150	COMMA
151	Args (9)
152	Exp (9)
153	ID: start
154	COMMA

```

155             Args (9)
156             Exp (9)
157             ID: mid
158             COMMA
159             Args (9)
160             Exp (9)
161             ID: end
162             RP
163             SEMI
164             RC
165             RC
166 ExtDefList (13)
167     ExtDef (13)
168         Specifier (13)
169             TYPE: int
170         FunDec (13)
171             ID: Merge
172             LP
173             VarList (13)
174                 ParamDec (13)
175                     Specifier (13)
176                         TYPE: int
177                     VarDec (13)
178                         VarDec (13)
179                             ID: arr
180                             LB
181                             INT: 100
182                             RB
183                     COMMA
184                     VarList (13)
185                         ParamDec (13)
186                             Specifier (13)

```

```

187         TYPE: int
188     VarDec (13)
189         VarDec (13)
190             ID: tmp
191         LB
192         INT: 100
193         RB
194     COMMA
195     VarList (13)
196         ParamDec (13)
197             Specifier (13)
198                 TYPE: int
199                 VarDec (13)
200                     ID: start
201             COMMA
202             VarList (13)
203                 ParamDec (13)
204                     Specifier (13)
205                         TYPE: int
206                         VarDec (13)
207                             ID: mid
208                 COMMA
209                 VarList (13)
210                     ParamDec (13)
211                         Specifier (13)
212                             TYPE: int
213                             VarDec (13)
214                                 ID: end
215     RP
216 CompSt (14)
217     LC
218     DefList (15)

```

```

219      Def (15)
220          Specifier (15)
221              TYPE: int
222      DecList (15)
223          Dec (15)
224              VarDec (15)
225                  ID: i
226                  ASSIGNOP
227                  Exp (15)
228                      ID: start
229      COMMA
230      DecList (15)
231          Dec (15)
232              VarDec (15)
233                  ID: j
234                  ASSIGNOP
235                  Exp (15)
236                      Exp (15)
237                          ID: mid
238                          PLUS
239                          Exp (15)
240                              INT: 1
241      COMMA
242      DecList (15)
243          Dec (15)
244              VarDec (15)
245                  ID: k
246                  ASSIGNOP
247                  Exp (15)
248                      ID: start
249      SEMI
250      StmtList (16)

```

```

251      Stmt (16)
252      WHILE
253      LP
254      Exp (16)
255      Exp (16)
256      Exp (16)
257      ID: i
258      RELOP
259      Exp (16)
260      Exp (16)
261      ID: mid
262      PLUS
263      Exp (16)
264      INT: 1
265      AND
266      Exp (16)
267      Exp (16)
268      ID: j
269      RELOP
270      Exp (16)
271      Exp (16)
272      ID: end
273      PLUS
274      Exp (16)
275      INT: 1
276      RP
277      Stmt (17)
278      CompSt (17)
279      LC
280      StmtList (18)
281      Stmt (18)
282      IF

```

283	LP
284	Exp (18)
285	Exp (18)
286	Exp (18)
287	ID: arr
288	LB
289	Exp (18)
290	ID: i
291	RB
292	RELOP
293	Exp (18)
294	Exp (18)
295	ID: arr
296	LB
297	Exp (18)
298	ID: j
299	RB
300	RP
301	Stmt (18)
302	CompSt (18)
303	LC
304	StmtList (19)
305	Stmt (19)
306	Exp (19)
307	Exp (19)
308	Exp (19)
309	ID: tmp
310	LB
311	Exp (19)
312	ID: k
313	RB
314	ASSIGNOP

315	Exp (19)
316	Exp (19)
317	ID: arr
318	LB
319	Exp (19)
320	ID: j
321	RB
322	SEMI
323	StmtList (20)
324	Stmt (20)
325	Exp (20)
326	Exp (20)
327	ID: k
328	ASSIGNOP
329	Exp (20)
330	Exp (20)
331	ID: k
332	PLUS
333	Exp (20)
334	INT: 1
335	SEMI
336	StmtList (21)
337	Stmt (21)
338	Exp (21)
339	Exp (21)
340	ID: j
341	ASSIGNOP
342	Exp (21)
343	Exp (21)
344	ID: j
345	PLUS
346	Exp (21)

347	INT: 1
348	SEMI
349	RC
350	ELSE
351	Stmt (23)
352	CompSt (23)
353	LC
354	StmtList (24)
355	Stmt (24)
356	Exp (24)
357	Exp (24)
358	Exp (24)
359	ID: tmp
360	LB
361	Exp (24)
362	ID: k
363	RB
364	ASSIGNOP
365	Exp (24)
366	Exp (24)
367	ID: arr
368	LB
369	Exp (24)
370	ID: i
371	RB
372	SEMI
373	StmtList (25)
374	Stmt (25)
375	Exp (25)
376	Exp (25)
377	ID: k
378	ASSIGNOP



379	Exp (25)
380	Exp (25)
381	ID: k
382	PLUS
383	Exp (25)
384	INT: 1
385	SEMI
386	StmtList (26)
387	Stmt (26)
388	Exp (26)
389	Exp (26)
390	ID: i
391	ASSIGNOP
392	Exp (26)
393	Exp (26)
394	ID: i
395	PLUS
396	Exp (26)
397	INT: 1
398	SEMI
399	RC
400	RC
401	StmtList (29)
402	Stmt (29)
403	WHILE
404	LP
405	Exp (29)
406	Exp (29)
407	ID: i
408	RELOP
409	Exp (29)
410	Exp (29)

```

411         ID: mid
412     PLUS
413     Exp (29)
414     INT: 1
415 RP
416 Stmt (29)
417   CompSt (29)
418     LC
419     StmtList (30)
420       Stmt (30)
421         Exp (30)
422           Exp (30)
423             Exp (30)
424               ID: tmp
425             LB
426               Exp (30)
427                 ID: k
428             RB
429           ASSIGNOP
430             Exp (30)
431               Exp (30)
432                 ID: arr
433             LB
434               Exp (30)
435                 ID: i
436             RB
437           SEMI
438     StmtList (31)
439       Stmt (31)
440         Exp (31)
441           Exp (31)
442             ID: k

```

```

443             ASSIGNOP
444             Exp (31)
445             Exp (31)
446             ID: k
447             PLUS
448             Exp (31)
449             INT: 1
450             SEMI
451 StmtList (32)
452 Stmt (32)
453 Exp (32)
454 Exp (32)
455 ID: i
456 ASSIGNOP
457 Exp (32)
458 Exp (32)
459 ID: i
460 PLUS
461 Exp (32)
462 INT: 1
463 SEMI
464 RC
465 StmtList (34)
466 Stmt (34)
467 WHILE
468 LP
469 Exp (34)
470 Exp (34)
471 ID: j
472 RELOP
473 Exp (34)
474 Exp (34)

```

475	ID: end
476	PLUS
477	Exp (34)
478	INT: 1
479	RP
480	Stmt (34)
481	CompSt (34)
482	LC
483	StmtList (35)
484	Stmt (35)
485	Exp (35)
486	Exp (35)
487	Exp (35)
488	ID: tmp
489	LB
490	Exp (35)
491	ID: k
492	RB
493	ASSIGNOP
494	Exp (35)
495	Exp (35)
496	ID: arr
497	LB
498	Exp (35)
499	ID: j
500	RB
501	SEMI
502	StmtList (36)
503	Stmt (36)
504	Exp (36)
505	Exp (36)
506	ID: k

507	ASSIGNOP
508	Exp (36)
509	Exp (36)
510	ID: k
511	PLUS
512	Exp (36)
513	INT: 1
514	SEMI
515	StmtList (37)
516	Stmt (37)
517	Exp (37)
518	Exp (37)
519	ID: j
520	ASSIGNOP
521	Exp (37)
522	Exp (37)
523	ID: j
524	PLUS
525	Exp (37)
526	INT: 1
527	SEMI
528	RC
529	StmtList (39)
530	Stmt (39)
531	Exp (39)
532	Exp (39)
533	ID: i
534	ASSIGNOP
535	Exp (39)
536	ID: start
537	SEMI
538	StmtList (40)

539	Stmt (40)
540	WHILE
541	LP
542	Exp (40)
543	Exp (40)
544	ID: i
545	RELOP
546	Exp (40)
547	ID: end
548	RP
549	Stmt (40)
550	CompSt (40)
551	LC
552	StmtList (41)
553	Stmt (41)
554	Exp (41)
555	Exp (41)
556	Exp (41)
557	ID: arr
558	LB
559	Exp (41)
560	ID: i
561	RB
562	ASSIGNOP
563	Exp (41)
564	Exp (41)
565	ID: tmp
566	LB
567	Exp (41)
568	ID: i
569	RB
570	SEMI

571	StmtList (42)
572	Stmt (42)
573	Exp (42)
574	Exp (42)
575	ID: i
576	ASSIGNOP
577	Exp (42)
578	Exp (42)
579	ID: i
580	PLUS
581	Exp (42)
582	INT: 1
583	SEMI
584	RC
585	RC

## 4 D 组测试用例

本组测试用例共 3 个，针对不同分组进行测试。对应分组的同学需要输出语法树，提示错误则不得分；其他分组的同学只需要在对应位置提示错误即可，如果打印了语法树，则将视为违规，将会倒扣分。

### 4.1 D-1

输入

```

1  int main() {
2      int a = 0x33;
3      int b = 100;
4      int c = 015;
5
6      if(a > b){
7          c = c + 0xa5;
8      }

```

```

9      else{
10         c = c + 077;
11     }
12
13     return c;
14 }

```

## 输出

```

1 Program (1)
2 ExtDefList (1)
3   ExtDef (1)
4     Specifier (1)
5       TYPE: int
6     FunDec (1)
7       ID: main
8       LP
9       RP
10    CompSt (1)
11      LC
12      DefList (2)
13        Def (2)
14          Specifier (2)
15            TYPE: int
16          DecList (2)
17            Dec (2)
18              VarDec (2)
19                ID: a
20                ASSIGNOP
21                Exp (2)
22                  INT: 51
23              SEMI
24            DefList (3)
25              Def (3)

```



```

26         Specifier (3)
27             TYPE: int
28         DecList (3)
29             Dec (3)
30                 VarDec (3)
31                     ID: b
32                     ASSIGNOP
33                     Exp (3)
34                         INT: 100
35             SEMI
36         DefList (4)
37             Def (4)
38                 Specifier (4)
39                     TYPE: int
40                 DecList (4)
41                     Dec (4)
42                         VarDec (4)
43                             ID: c
44                             ASSIGNOP
45                             Exp (4)
46                                 INT: 13
47             SEMI
48         StmtList (6)
49             Stmt (6)
50                 IF
51                 LP
52                 Exp (6)
53                     Exp (6)
54                         ID: a
55                     RELOP
56                     Exp (6)
57                         ID: b

```

```

58      RP
59      Stmt (6)
60      CompSt (6)
61      LC
62      StmtList (7)
63      Stmt (7)
64      Exp (7)
65      Exp (7)
66      ID: c
67      ASSIGNOP
68      Exp (7)
69      Exp (7)
70      ID: c
71      PLUS
72      Exp (7)
73      INT: 165
74      SEMI
75      RC
76  ELSE
77  Stmt (9)
78      CompSt (9)
79      LC
80      StmtList (10)
81      Stmt (10)
82      Exp (10)
83      Exp (10)
84      ID: c
85      ASSIGNOP
86      Exp (10)
87      Exp (10)
88      ID: c
89      PLUS

```

```

90             Exp (10)
91             INT: 63
92             SEMI
93             RC
94         StmtList (13)
95         Stmt (13)
96         RETURN
97         Exp (13)
98         ID: c
99         SEMI
100     RC

```

说明：1.1 分组的同学需要输出该语法树，8 进制和 16 进制数必须正确转换；其他分组的同学只要提示相应的错误（不输出语法树即可）。

## 4.2 D-2

输入

```

1  int main() {
2      float f_1 = 0.12345;
3      float f_2 = 0.23e5;
4      float f_3 = .13E+6;
5      float f_4 = 12.e-10;
6      float f_5 = 01.5E-11;
7  }

```

输出

```

1  Program (1)
2  ExtDefList (1)
3      ExtDef (1)
4          Specifier (1)
5              TYPE: int
6          FunDec (1)
7              ID: main

```

```

8      LP
9      RP
10     CompSt (1)
11     LC
12     DefList (2)
13     Def (2)
14     Specifier (2)
15     TYPE: float
16     Declist (2)
17     Dec (2)
18     VarDec (2)
19     ID: f_1
20     ASSIGNOP
21     Exp (2)
22     FLOAT: 0.123450
23     SEMI
24     DefList (3)
25     Def (3)
26     Specifier (3)
27     TYPE: float
28     Declist (3)
29     Dec (3)
30     VarDec (3)
31     ID: f_2
32     ASSIGNOP
33     Exp (3)
34     FLOAT: 23000.000000
35     SEMI
36     DefList (4)
37     Def (4)
38     Specifier (4)
39     TYPE: float

```

```

40         DecList (4)
41         Dec (4)
42         VarDec (4)
43         ID: f_3
44         ASSIGNOP
45         Exp (4)
46         FLOAT: 130000.000000
47     SEMI
48 DefList (5)
49     Def (5)
50         Specifier (5)
51         TYPE: float
52         DecList (5)
53         Dec (5)
54         VarDec (5)
55         ID: f_4
56         ASSIGNOP
57         Exp (5)
58         FLOAT: 0.000000
59     SEMI
60 DefList (6)
61     Def (6)
62         Specifier (6)
63         TYPE: float
64         DecList (6)
65         Dec (6)
66         VarDec (6)
67         ID: f_5
68         ASSIGNOP
69         Exp (6)
70         FLOAT: 0.000000
71     SEMI

```

说明：1.2 分组的同学需要输出语法树，注意科学计数法浮点数的正确转换。其它分组同学只需要提示相应错误（不输出语法树）即可。

### 4.3 D-3

输入

```
1 // This is the one line comment.
2
3 /*
4  * Traverse all elements in the array and find the max value
5  */
6
7 int find_max(int array[100]){
8     int i = 0;
9     int maxm = 0;
10    while(i < 100){
11        if(maxm > array[i]){ /* compare with current maxm */
12            maxm = array[i];
13        }
14        i = i + 1; /* step further */
15    }
16 }
```

输出

```
1 Program (7)
2 ExtDefList (7)
3   ExtDef (7)
4     Specifier (7)
5       TYPE: int
6     FunDec (7)
7       ID: find_max
8       LP
```

```

9      VarList (7)
10     ParamDec (7)
11     Specifier (7)
12     TYPE: int
13     VarDec (7)
14     VarDec (7)
15     ID: array
16     LB
17     INT: 100
18     RB
19     RP
20 CompSt (7)
21 LC
22 DefList (8)
23 Def (8)
24 Specifier (8)
25 TYPE: int
26 DecList (8)
27 Dec (8)
28 VarDec (8)
29 ID: i
30 ASSIGNOP
31 Exp (8)
32 INT: 0
33 SEMI
34 DefList (9)
35 Def (9)
36 Specifier (9)
37 TYPE: int
38 DecList (9)
39 Dec (9)
40 VarDec (9)

```

```

41         ID: maxm
42     ASSIGNOP
43     Exp (9)
44     INT: 0
45     SEMI
46 StmtList (10)
47     Stmt (10)
48     WHILE
49     LP
50     Exp (10)
51     Exp (10)
52     ID: i
53     RELOP
54     Exp (10)
55     INT: 100
56     RP
57     Stmt (10)
58     CompSt (10)
59     LC
60     StmtList (11)
61     Stmt (11)
62     IF
63     LP
64     Exp (11)
65     Exp (11)
66     ID: maxm
67     RELOP
68     Exp (11)
69     Exp (11)
70     ID: array
71     LB
72     Exp (11)

```



73	ID: i
74	RB
75	RP
76	Stmt (11)
77	CompSt (11)
78	LC
79	StmtList (12)
80	Stmt (12)
81	Exp (12)
82	Exp (12)
83	ID: maxm
84	ASSIGNOP
85	Exp (12)
86	Exp (12)
87	ID: array
88	LB
89	Exp (12)
90	ID: i
91	RB
92	SEMI
93	RC
94	StmtList (14)
95	Stmt (14)
96	Exp (14)
97	Exp (14)
98	ID: i
99	ASSIGNOP
100	Exp (14)
101	Exp (14)
102	ID: i
103	PLUS
104	Exp (14)

105			INT: 1
106		SEMI	
107		RC	
108	RC		

说明：1.3 分组的同学需要输出语法树，不能提示有语法错误；其他分组同学只需要提示相应错误（不输出语法树）即可。

## 5 E 组测试用例

本组测试用例共 6 个，针对不同分组进行测试。

### 5.1 E1-1

这组测试用例针对 1.1 分组的同学。

输入

```

1  int Octal() {
2      int o1 = 09888;
3      int o2 = 01234567;
4      int o3 = o2 + o1 + 0526454;
5  }
6
7  int Hex() {
8      int h1 = 0xffffffff;
9      int h2 = -0x14daf;
10     int h3 = h1 + h2 + 0xggg;
11 }
```

输出

```

1 Error type A at Line 2: Illegal octal number '09888'
2 Error type A at Line 10: Illegal hexadecimal number '0xggg'
```

说明：仅 1.1 分组的同学需要测试这个用例，这两处错误都可以识别成错误 B。

## 5.2 E1-2

这组测试用例针对 1.1 分组的同学。

输入

```
1  int a;
2  int N;
3  float file[500];
4
5  struct Type myfunc1() {}
6
7  int myfunc2() {
8      int k;
9      int length=strlen(file);
10     int spaces=012346;
11     while(k < length){
12         if(file[k] == a){
13             spaces = spaces + 1;
14         }
15     }
16
17     if(0xabcdef > 01234567){
18         spaces = spaces + 1;
19     }
20     else{
21         spaces = spaces - 1;
22     }
23 }
```

输出

```
1 Program (1)
2 ExtDefList (1)
3     ExtDef (1)
4         Specifier (1)
```

```

5      TYPE: int
6      ExtDecList (1)
7      VarDec (1)
8      ID: a
9      SEMI
10     ExtDefList (2)
11     ExtDef (2)
12     Specifier (2)
13     TYPE: int
14     ExtDecList (2)
15     VarDec (2)
16     ID: N
17     SEMI
18     ExtDefList (3)
19     ExtDef (3)
20     Specifier (3)
21     TYPE: float
22     ExtDecList (3)
23     VarDec (3)
24     VarDec (3)
25     ID: file
26     LB
27     INT: 500
28     RB
29     SEMI
30     ExtDefList (5)
31     ExtDef (5)
32     Specifier (5)
33     StructSpecifier (5)
34     STRUCT
35     Tag (5)
36     ID: Type

```

```

37     FunDec (5)
38         ID: myfunc1
39         LP
40         RP
41     CompSt (5)
42         LC
43         RC
44 ExtDefList (7)
45     ExtDef (7)
46         Specifier (7)
47             TYPE: int
48         FunDec (7)
49             ID: myfunc2
50             LP
51             RP
52     CompSt (7)
53         LC
54         DefList (8)
55             Def (8)
56                 Specifier (8)
57                     TYPE: int
58                 DecList (8)
59                     Dec (8)
60                         VarDec (8)
61                             ID: k
62                             SEMI
63                 DefList (9)
64                     Def (9)
65                         Specifier (9)
66                             TYPE: int
67                         DecList (9)
68                             Dec (9)

```

```

69         VarDec (9)
70             ID: length
71         ASSIGNOP
72         Exp (9)
73             ID: strlen
74             LP
75             Args (9)
76                 Exp (9)
77                     ID: file
78             RP
79         SEMI
80     DefList (10)
81         Def (10)
82             Specifier (10)
83                 TYPE: int
84             Declist (10)
85                 Dec (10)
86                     VarDec (10)
87                         ID: spaces
88                     ASSIGNOP
89                     Exp (10)
90                         INT: 5350
91         SEMI
92 StmtList (11)
93     Stmt (11)
94         WHILE
95             LP
96             Exp (11)
97                 Exp (11)
98                     ID: k
99             RELOP
100            Exp (11)

```

101	ID: length
102	RP
103	Stmt (11)
104	CompSt (11)
105	LC
106	StmtList (12)
107	Stmt (12)
108	IF
109	LP
110	Exp (12)
111	Exp (12)
112	Exp (12)
113	ID: file
114	LB
115	Exp (12)
116	ID: k
117	RB
118	RELOP
119	Exp (12)
120	ID: a
121	RP
122	Stmt (12)
123	CompSt (12)
124	LC
125	StmtList (13)
126	Stmt (13)
127	Exp (13)
128	Exp (13)
129	ID: spaces
130	ASSIGNOP
131	Exp (13)
132	Exp (13)

133		ID: spaces
134		PLUS
135		Exp (13)
136		INT: 1
137		SEMI
138		RC
139	RC	
140	StmtList (17)	
141	Stmt (17)	
142	IF	
143	LP	
144	Exp (17)	
145	Exp (17)	
146	INT: 11259375	
147	RELOP	
148	Exp (17)	
149	INT: 342391	
150	RP	
151	Stmt (17)	
152	CompSt (17)	
153	LC	
154	StmtList (18)	
155	Stmt (18)	
156	Exp (18)	
157	Exp (18)	
158	ID: spaces	
159	ASSIGNOP	
160	Exp (18)	
161	Exp (18)	
162	ID: spaces	
163	PLUS	
164	Exp (18)	



```

165             INT: 1
166         SEMI
167     RC
168 ELSE
169     Stmt (20)
170     CompSt (20)
171     LC
172     StmtList (21)
173     Stmt (21)
174     Exp (21)
175     Exp (21)
176     ID: spaces
177     ASSIGNOP
178     Exp (21)
179     Exp (21)
180     ID: spaces
181     MINUS
182     Exp (21)
183     INT: 1
184     SEMI
185 RC
186 RC

```

说明：仅 1.1 分组的同学需要测试这个用例，并输出语法树。

### 5.3 E2-1

这组测试用例针对 1.2 分组的同学。

输入

```

1  float float_error(){
2      float f1 = 0.114514e;
3      float f2 = 123.1e+10;
4      float f3 = f1 + f2 + 127.2.2.1;
5  }

```

输出

```
1 Error type A at line 2: Invalid floating point number
2 Error type A at line 4: Invalid floating point number
```

说明：仅 1.2 分组的同学需要测试这个用例，这里的两个错误都可以识别成 B 类错误。

## 5.4 E2-2

这组测试用例针对 1.2 分组的同学。

输入

```
1 float right_floats() {
2     float f1 = e1.e1;
3     float f2 = -100.4e+100;
4     float f3 = f1 / 0.5e-1;
5     float f4 = f2 * .51e+10;
6 }
```

输出

```
1 Program (1)
2 ExtDefList (1)
3     ExtDef (1)
4         Specifier (1)
5             TYPE: float
6         FunDec (1)
7             ID: right_floats
8             LP
9             RP
10        CompSt (1)
11            LC
12        DefList (2)
13            Def (2)
14                Specifier (2)
15                    TYPE: float
```

```

16      DecList (2)
17      Dec (2)
18      VarDec (2)
19      ID: f1
20      ASSIGNOP
21      Exp (2)
22      Exp (2)
23      ID: e1
24      DOT
25      ID: e1
26      SEMI
27  DefList (3)
28  Def (3)
29  Specifier (3)
30  TYPE: float
31  DecList (3)
32  Dec (3)
33  VarDec (3)
34  ID: f2
35  ASSIGNOP
36  Exp (3)
37  MINUS
38  Exp (3)
39  FLOAT: inf
40  SEMI
41  DefList (4)
42  Def (4)
43  Specifier (4)
44  TYPE: float
45  DecList (4)
46  Dec (4)
47  VarDec (4)

```

```

48         ID: f3
49     ASSIGNOP
50     Exp (4)
51         Exp (4)
52             ID: f1
53         DIV
54         Exp (4)
55             FLOAT: 0.050000
56     SEMI
57 DefList (5)
58     Def (5)
59         Specifier (5)
60             TYPE: float
61         DecList (5)
62             Dec (5)
63                 VarDec (5)
64                     ID: f4
65                 ASSIGNOP
66                 Exp (5)
67                     Exp (5)
68                         ID: f2
69                 STAR
70                 Exp (5)
71                     FLOAT: 5100000256.000000
72     SEMI
73 RC

```

说明：仅 1.2 分组的同学需要测试这个用例，并输出语法树。

## 5.5 E3-1

这组测试用例针对 1.3 分组的同学。

输入

```
1  /* sjfgkjghkxjhgudfh */
```

```

2
3  /*
4  #include <stdio.h>
5  #include <unistd.h>
6  #include <string.h>
7
8  #define LOG(format, ...) printf(format,##__VA_ARGS__)
9
10 void usage(char *program)
11 {
12     LOG("Usage:\n%s [-i interface [null/eth0/eth1...]] [-p
        listenport] [-h help] \n", program);
13     LOG("example:\n");
14     LOG("\t%s -i eth0 -p 6000 \n",program);
15 }
16
17 int main(int argc, char *argv[])
18 {
19     int c;
20     char interface[64];
21     int listen_port;
22     usage(argv[0]);
23
24     LOG("You have input %d params\n",argc-1);
25     while ((c = getopt(argc, argv, "i:p:h?")) != -1) {
26         switch (c) {
27             case 'i':
28                 strcpy(interface , optarg);
29                 LOG("interface %s \n",interface);
30                 break;
31             case 'p':
32                 listen_port = atoi(optarg);

```

```

33         LOG("listen_port %d \n", listen_port);
34         break;
35     case 'h':
36     case '?':
37         LOG("Need help ?\n");
38         break;
39     default:
40         usage(argv[0]);
41         return 1;
42     }
43 }
44
45     return 0;
46 }
47 */
48
49 int main(){
50     hello();
51     error_func(int b);
52 }

```

输出

```

1 Error type B at line 51: Unexpected 'int'.

```

说明：仅 1.3 分组的同学需要测试这个用例。

## 5.6 E3-2

这组测试用例针对 1.3 分组的同学。

输入

```

1  /**comment
2  still
3  */
4  /* ////\asfdlkajhsldf\\\<>@#&

```

```

5  */
6  int main() //a function
7  {
8      float h, s;
9      int i;
10     h = s = 100;
11     h = h / 2; // first
12     i = 2;
13     /* a new comment*/
14     while (i <= 10) //\\//\\//\\*
15     {
16         s = s + 2 * h;
17         h = h / 2 /*\\//\\//\\*\\//\\*****\\*/;
18         i = i + 1;
19     }
20     return 0;
21 }

```

## 输出

```

1 Program (6)
2 ExtDefList (6)
3   ExtDef (6)
4     Specifier (6)
5       TYPE: int
6     FunDec (6)
7       ID: main
8       LP
9       RP
10    CompSt (7)
11      LC
12      DefList (8)
13        Def (8)
14          Specifier (8)

```

```

15         TYPE: float
16     DecList (8)
17         Dec (8)
18             VarDec (8)
19                 ID: h
20         COMMA
21     DecList (8)
22         Dec (8)
23             VarDec (8)
24                 ID: s
25     SEMI
26 DefList (9)
27     Def (9)
28         Specifier (9)
29             TYPE: int
30         DecList (9)
31             Dec (9)
32                 VarDec (9)
33                     ID: i
34     SEMI
35 StmtList (10)
36     Stmt (10)
37         Exp (10)
38             Exp (10)
39                 ID: h
40         ASSIGNOP
41         Exp (10)
42             Exp (10)
43                 ID: s
44         ASSIGNOP
45         Exp (10)
46             INT: 100

```



```

47         SEMI
48     StmtList (11)
49     Stmt (11)
50         Exp (11)
51             Exp (11)
52                 ID: h
53             ASSIGNOP
54             Exp (11)
55                 Exp (11)
56                     ID: h
57                 DIV
58                 Exp (11)
59                     INT: 2
60     SEMI
61     StmtList (12)
62     Stmt (12)
63         Exp (12)
64             Exp (12)
65                 ID: i
66             ASSIGNOP
67             Exp (12)
68                 INT: 2
69     SEMI
70     StmtList (14)
71     Stmt (14)
72         WHILE
73         LP
74         Exp (14)
75             Exp (14)
76                 ID: i
77             RELOP
78             Exp (14)

```

79	INT: 10
80	RP
81	Stmt (15)
82	CompSt (15)
83	LC
84	StmtList (16)
85	Stmt (16)
86	Exp (16)
87	Exp (16)
88	ID: s
89	ASSIGNOP
90	Exp (16)
91	Exp (16)
92	ID: s
93	PLUS
94	Exp (16)
95	Exp (16)
96	INT: 2
97	STAR
98	Exp (16)
99	ID: h
100	SEMI
101	StmtList (17)
102	Stmt (17)
103	Exp (17)
104	Exp (17)
105	ID: h
106	ASSIGNOP
107	Exp (17)
108	Exp (17)
109	ID: h
110	DIV

```

111             Exp (17)
112             INT: 2
113         SEMI
114     StmtList (18)
115     Stmt (18)
116     Exp (18)
117     Exp (18)
118     ID: i
119     ASSIGNOP
120     Exp (18)
121     Exp (18)
122     ID: i
123     PLUS
124     Exp (18)
125     INT: 1
126     SEMI
127     RC
128 StmtList (20)
129 Stmt (20)
130     RETURN
131     Exp (20)
132     INT: 0
133     SEMI
134 RC

```

说明：仅 1.3 分组的同学需要测试这个用例，需要输出正确的语法树。

## 6 结束语

如果对本测试用例有任何疑议，可以写邮件与张灵毓助教联系，注意同时抄送给许老师。