

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  string int_to_string(int a){
5      stringstream ss;
6      ss << a;
7      string str = ss.str();
8      return str;
9  }
10
11 vector<string> number_lines(vector<string>sp){
12     int flag = 0;
13     string s;
14
15     int flag3 = -1;
16     for(int i=0;i<sp.size();i++){
17         s = "";
18         int sz = sp[i].size();
19         flag3 = -1;
20         for(int j=0;j<sz;j++) if(sp[i][j]=='\t') sp[i][j] = ' ';
21         for(int j=0;j<sz;j++){
22             if(j!=sz-1 && sp[i][j]!=' ' && sp[i][j+1]!=' '){ s = s + sp[i][j] + ' ';
23             else if(sp[i][j]!=' ') s += sp[i][j];
24         }
25         for(int j=0;j<sz;j++){
26             if(sp[i][j]==' '){
27                 flag3 = j;
28                 break;
29             }
30         }
31         if(flag3!=-1){
32             string p = "";
33             for(int j=0;s[j]!='';j++) p += s[j];
34             p += "\n";
35             for(int j=flag3+1,r=0;sp[i][j]!='';j++) p += sp[i][j];
36             for(int j=0,r=0;j<s.size();j++){
37                 if(s[j]=='') r++;
38                 if(r==2) p +=s[j];
39             }
40             swap(s,p);
41         }
42         swap(sp[i],s);
43     }
44
45     vector<string>sp1;
46
47     int flag1 = 0,flag2=0;
48     for(int i=0;i<sp.size();i++){
49         string str = int_to_string(i+1);
50         int sz = sp[i].size();
51         if(sz==0){
52             sp1.push_back(str);
53             continue;
54         }
55         for(int j=0;j<sz;j++){
56             if(j!=sz-1 && sp[i][j]=='/' && sp[i][j+1]=='/'){
57                 flag1 = 1;
58                 for(int k=0;k<j;k++){
59                     cout<<sp[i][k];
60                     cerr<<sp[i][k];
61                 }
62                 break;
63             }
64             if(j!=sz-1 && sp[i][j]=='/' && sp[i][j+1]=='*'){
65                 flag2 = 1;
66                 for(int k=0;k<j;k++){

```

```

67         cout<<sp[i][k];
68         cerr<<sp[i][k];
69     }
70 }
71 if(j!=sz-1 && sp[i][j]=='*' && sp[i][j+1]=='/'){
72     flag2 = 0;
73     flag1 = 1;
74     break;
75 }
76 }
77 if(flag1){
78     flag1 = 0;
79     spl.push_back(str);
80     continue;
81 }
82 if(flag2){
83     spl.push_back(str);
84     continue;
85 }
86 str = str + " " + sp[i];
87 spl.push_back(str);
88 }
89
90 return spl;
91
92 }
93
94 vector<string> paranthesis_error(vector<string> sp){
95
96     stack<int>st;
97     vector<string>err;
98
99     for(int i=0;i<sp.size();i++){
100         for(int j=0;j<sp[i].size();j++){
101             if(sp[i][j]=='{') st.push(i+1);
102             else if(sp[i][j]=='}'){
103                 if( !st.empty() ) st.pop();
104                 else err.push_back("Error: Misplaced '}' at line "+int_to_string(i+1));
105             }
106         }
107     }
108
109     if( !st.empty() ) err.push_back("Error: Not Balanced Parentheses at line "+int_to_string(sp.size()));
110
111     return err;
112 }
113
114
115 vector<string> if_else_error(vector<string> sp){
116
117     bool ok = false;
118     vector<string>err;
119     int sz = sp.size();
120     for(int i=0;i<sz;i++){
121         if(sz<4)continue;
122         int x = sp[i].size();
123         for(int j=0;j<x;j++){
124             if(j+1<x && sp[i][j]=='i' && sp[i][j+1]=='f') ok = true;
125             if(j+3<x && sp[i][j]=='e' && sp[i][j+1]=='l' && sp[i][j+2]=='s' && sp[i][j+3]=='e'){
126                 if( ok ){
127                     ok = false;
128                     continue;
129                 }
130                 else err.push_back("Error: Not Matched else at line "+int_to_string(i+1));
131             }
132         }
133     }

```

```

133     }
134
135     return err;
136 }
137
138 bool comp(char a){
139     if(a=='=' || a=='>' || a=='<' ) return false;
140
141     return true;
142 }
143
144 bool col(char a){
145
146     if(a=='.' || a==',' || a=='+' || a=='-' || a=='*' || a=='/' || a=='(' || a==')' || a=='\'' || a=='\"') return true
147 ;
148     return false;
149 }
150
151 vector<string> dup_token_error(vector<string> sp){
152
153     vector<string>err;
154     int sz = sp.size();
155
156     for(int j=0;j<sz;j++){
157
158         string p = "",s=sp[j];
159
160         for(int i=0;i<s.size();i++){
161             if(col(s[i]) && col(s[i+1])==false) p = p+" "+s[i]+" ";
162             else if(col(s[i]) && col(s[i+1])) p = p+" "+s[i];
163             else p += s[i];
164         }
165
166         s = p[0];
167
168         for(int i=1;i<p.size()-1;i++){
169             if(p[i]=='=' && comp(p[i-1]) && comp(p[i+1])) s = s+" "+p[i]+" ";
170             else s +=p[i];
171         }
172
173         p = "";
174
175
176         for(int i=0;i<s.size();i++){
177             if(i!=s.size()-1 && s[i]!=' ' && s[i+1]==' ') p = p + s[i] + ' ';
178             else if(s[i]!=' ') p += s[i];
179         }
180
181         s = p[0];
182
183         for(int i=1;i<p.size()-1;i++){
184             if(comp(p[i])==false && comp(p[i+1])==false){
185                 s = s + " "+ p[i]+p[i+1] + " ";
186                 i++;
187             }
188             else s += p[i];
189         }
190
191
192         s+= p[p.size()-1];
193
194         istringstream ss(s);
195
196         string last = "";
197

```

```

198     while(ss>>s){
199         if(s==last) err.push_back("Error: Duplicate token at line "+int_to_string(j+1));
200         last = s;
201     }
202 }
203 }
204
205 return err;
206 }
207
208
209
210 int main(){
211
212     freopen("input.txt","r",stdin);
213     freopen("out.txt","w",stdout);
214
215     string s;
216
217     vector<string>sp, paran_error, if_else_err, dup_token_err, error;
218
219     cerr<<"input\n";
220
221     while(getline(cin,s)){
222         sp.push_back(s);
223         cerr<<s<<"\n";
224     }
225
226     cerr<<"\n";
227
228     sp = number_lines(sp);
229
230     cerr<<"\noutput:\n";
231
232     cerr<<"Recognized tokens in the lines of code:\n";
233
234     for(int i=0;i<sp.size();i++){
235         cout<<sp[i]<<"\n";
236         cerr<<sp[i]<<"\n";
237     }
238
239     paran_error = paranthesis_error(sp);
240
241     if_else_err = if_else_error(sp);
242
243     dup_token_err = dup_token_error(sp);
244
245     paran_error.erase( unique( paran_error.begin(), paran_error.end() ), paran_error.end() );
246
247     if_else_err.erase( unique( if_else_err.begin(), if_else_err.end() ), if_else_err.end() );
248
249     dup_token_err.erase( unique( dup_token_err.begin(), dup_token_err.end() ), dup_token_err.end() );
250
251
252     cout<<"\n\nERROR: \n";
253     cerr<<"\n\nERROR: \n";
254
255     for(int i=0;i<paran_error.size();i++){
256         cout<<paran_error[i]<<"\n";
257         cerr<<paran_error[i]<<"\n";
258     }
259
260     for(int i=0;i<if_else_err.size();i++){
261         cout<<if_else_err[i]<<"\n";
262         cerr<<if_else_err[i]<<"\n";
263     }

```

```
264
265     for(int i=0;i<dup_token_err.size();i++){
266         cout<<dup_token_err[i]<<"\n";
267         cerr<<dup_token_err[i]<<"\n";
268     }
269
270     return 0;
271 }
```