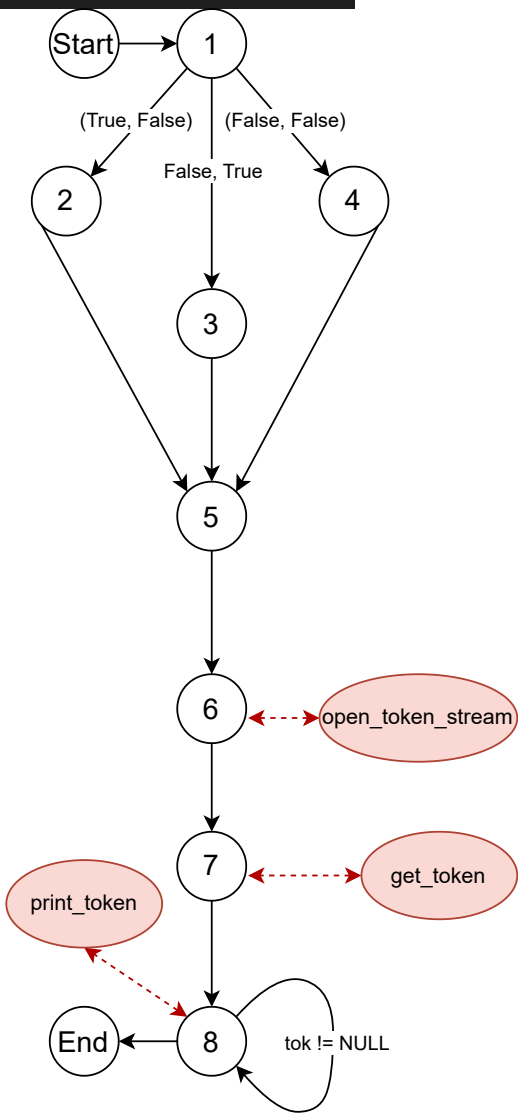
Name: Wilmer Soriano
ID: 1001885481

CFG - Printtokens

Main

```
458    public static void main(String[] args)  {
459        String fname = null;
460        if (args.length == 0) { /* if not given filename,take as '"'' */
461            fname = new String();
462        } else if (args.length == 1) {
463            fname = args[0];
464        } else {
465            System.out.print("Error! Please give the token stream\n");
466        }
467        Printtokens t = new Printtokens();
468        BufferedReader br = t.open_token_stream(fname); /* open token stream */
469        String tok = t.get_token(br);
470        while (tok != null) {   /* take one token each time until eof */
471            t.print_token(tok);
472            tok = t.get_token(br);
473
474        }
475    }
```

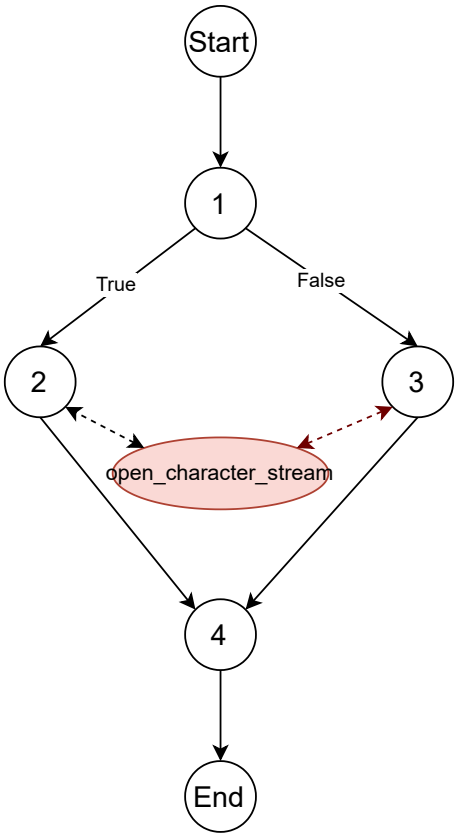
Block Number	Lines	Entry	Exit	Function Cells
1	459, 460	459	460	
2	461	461	461	
3	463	463	463	
4	465	465	465	
5	467	467	469	
6	468	468	468	open_token_stream
7	469	469	469	get_token
8	471,472	471	472	print_token



open_token_stream

```
70      /*****************************************************/
71      /* NAME:    open_token_stream                        */
72      /* INPUT:       a filename                           */
73      /* OUTPUT:      a BufferedReader                     */
74      /* DESCRIPTION: when filename is EMPTY,choice standard */
75      /*              input device as input source         */
76      /*****************************************************/
77      BufferedReader open_token_stream(String fname)
78      {
79          BufferedReader br;
80      if(fname==null || fname.equals(""))
81          br=open_character_stream(null);
82       else
83          br=open_character_stream(fname);
84       return br;
85      }
```

Block Number	Lines	Entry	Exit	Function Cells
1	79,80	79	80	
2	81	81	81	open_character_stream
3	83	83	83	open_character_stream
4	84	84	84	

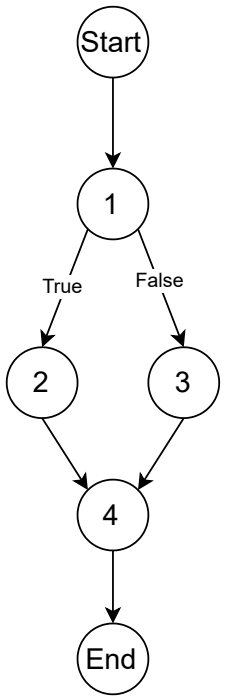


open_character_stream

```java
/*********************************************/
/* NMAE:    open_character_stream          */
/* INPUT:       a filename                  */
/* OUTPUT:      a BufferedReader */
/* DESCRIPTION: when not given a filename,     */
/*              open stdin,otherwise open      */
/*              the existed file               */
/*********************************************/
BufferedReader open_character_stream(String fname) {
    BufferedReader br = null;
    if (fname == null) {
        br = new BufferedReader(new InputStreamReader(System.in));
    } else {
        try {
            FileReader fr = new FileReader(fname);
            br = new BufferedReader(fr);
        } catch (FileNotFoundException e) {
            System.out.print("The file " + fname +" doesn't exists\n");
            e.printStackTrace();
        }
    }

    return br;
}
```

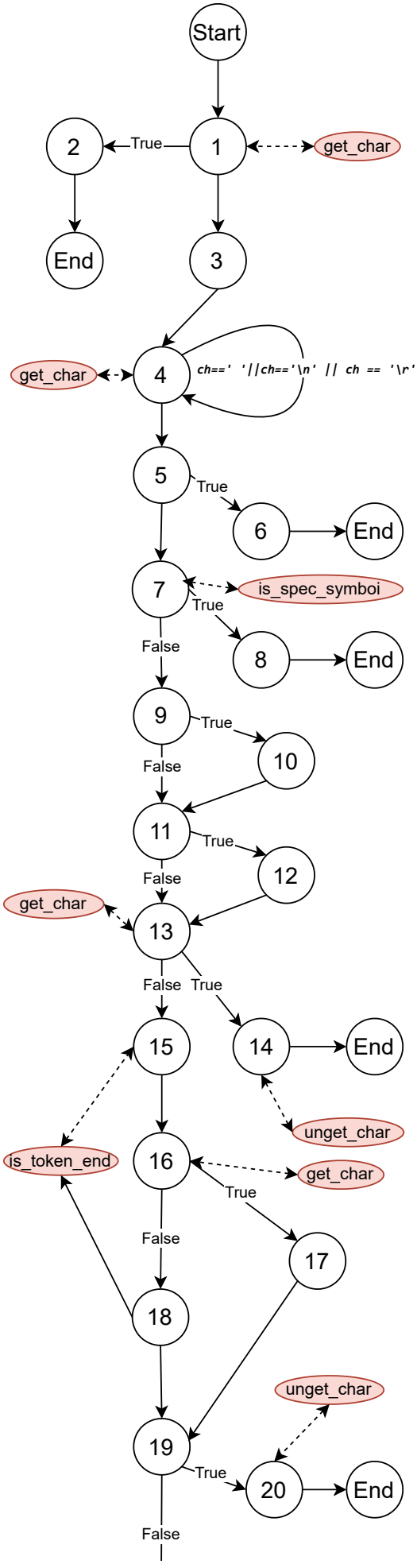
Block Number	Lines	Entry	Exit	Function Cells
1	23,24	23	24	
2	25	25	25	
3	28,29	28	29	
4	36	36	36	



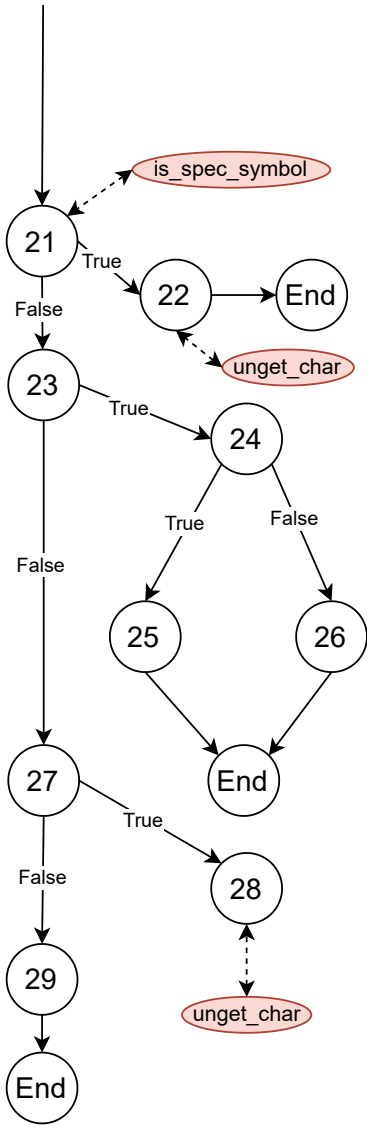
get_token

```
87    /*******************************************************/
88    /* NAME :   get_token                                  */
89    /* INPUT:   a BufferedReader          */
90    /* OUTPUT:      a token string                                 */
91    /* DESCRIPTION: according the syntax of tokens,dealing  */
92    /*              with different case  and get one token  */
93    /*******************************************************/
94    String get_token(BufferedReader br)
95    {
96      int i=0,j;
97      int id=0;
98      int res = 0;
99      char ch = '\0';
100
101     StringBuilder sb = new StringBuilder();
102
103      try {
104          res = get_char(br);
105          if (res == -1) {
106              return null;
107          }
108          ch = (char)res;
109         while(ch==' '||ch=='\n' || ch == '\r')
110         {
111           res = get_char(br);
112           ch = (char)res;
113         }
114
115      if(res == -1)return null;
116      sb.append(ch);
117      if(is_spec_symbol(ch)==true)return sb.toString();
118      if(ch =='"')id=2;    /* prepare for string */
119      if(ch ==59)id=1;    /* prepare for comment */
120
121      res = get_char(br);
122      if (res == -1) {
123          unget_char(ch,br);
124          return sb.toString();
125      }
126      ch = (char)res;
127
128       while (is_token_end(id,res) == false)/* until meet the end character */
129       {
130           sb.append(ch);
131           br.mark(4);
132           res = get_char(br);
133           if (res == -1) {
134               break;
135           }
136           ch = (char)res;
137       }
138
139       if(res == -1)       /* if end character is eof token    */
140          { unget_char(ch,br);        /* then put back eof on token_stream */
141            return sb.toString();
142          }
143
144       if(is_spec_symbol(ch)==true)     /* if end character is special_symbol */
145          { unget_char(ch,br);        /* then put back this character       */
146            return sb.toString();
147          }
148       if(id==1)                  /* if end character is " and is string */
149         {
150           if (ch == '"') {
151               sb.append(ch);
152           }
153           return sb.toString();
154         }
155       if(id==0 && ch==59)
156                                     /* when not in string or comment,meet ";" */
157         { unget_char(ch,br);       /* then put back this character         */
158           return sb.toString();
159         }
160     } catch (IOException e) {
161         e.printStackTrace();
162     }
163
164       return sb.toString();                 /* return nomal case token            */
165   }
```

Block Number	Lines	Entry	Exit	Function Cells
1	96,97,98,99, 101,104,105	96	105	get_char
2	106	106	106	
3	108,109	108	109	
4	111, 112	111	112	get_char
5	115a	115a	115a	
6	115b	115b	115b	
7	116, 117a	116	117a	is_spec_symbol
8	117b	117b	117b	
9	118a	118a	118a	
10	118b	118b	118b	
11	119a	119a	119a	
12	119b	119b	119b	
13	121,122	121	122	get_char
14	123,124	123	124	unget_char
15	126,128	126	128	is_token_end
16	130,131,132,133	130	133	get_char
17	134	134	134	
18	136	136	136	
19	139	139	139	
20	140,141	140	141	unget_char



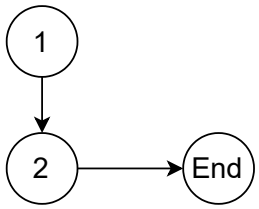
Block Number	Lines	Entry	Exit	Function Cells
21	144	144	144	is_spec_symbol
22	145,146	145	146	unget_char
23	148	148	148	
24	150	150	150	
25	151	151	151	
26	153	153	153	
27	155	155	155	
28	157,158	157	158	unget_char
29	164	164	164	



get_char

```
39      /*********************************************/
40      /* NAME:     get_char                        */
41      /* INPUT:        a BufferedReader      */
42      /* OUTPUT:       a character; when EOF, return -1            */
43      /*********************************************/
44      int get_char(BufferedReader br){
45          int ch = 0;
46          try {
47              br.mark(4);
48              ch= br.read();
49          } catch (IOException e) {
50              e.printStackTrace();
51          }
52          return ch;
53      }
```

Block Number	Lines	Entry	Exit	Function Cells
1	45, 47, 48	45	48	
2	52	52	52	

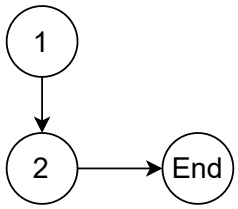


---

unget_char

```
55      /*************************************************/
56      /* NAME:       unget_char                        */
57      /* INPUT:      a BufferedReader,a character */
58      /* OUTPUT:     a character                       */
59      /* DESCRIPTION:move backward.when unable to put back,return -1(EOF)  */
60      /*************************************************/
61      char unget_char (int ch,BufferedReader br) {
62        try {
63          br.reset();
64      } catch (IOException e) {
65          e.printStackTrace();
66      }
67           return 0;
68      }
69
```

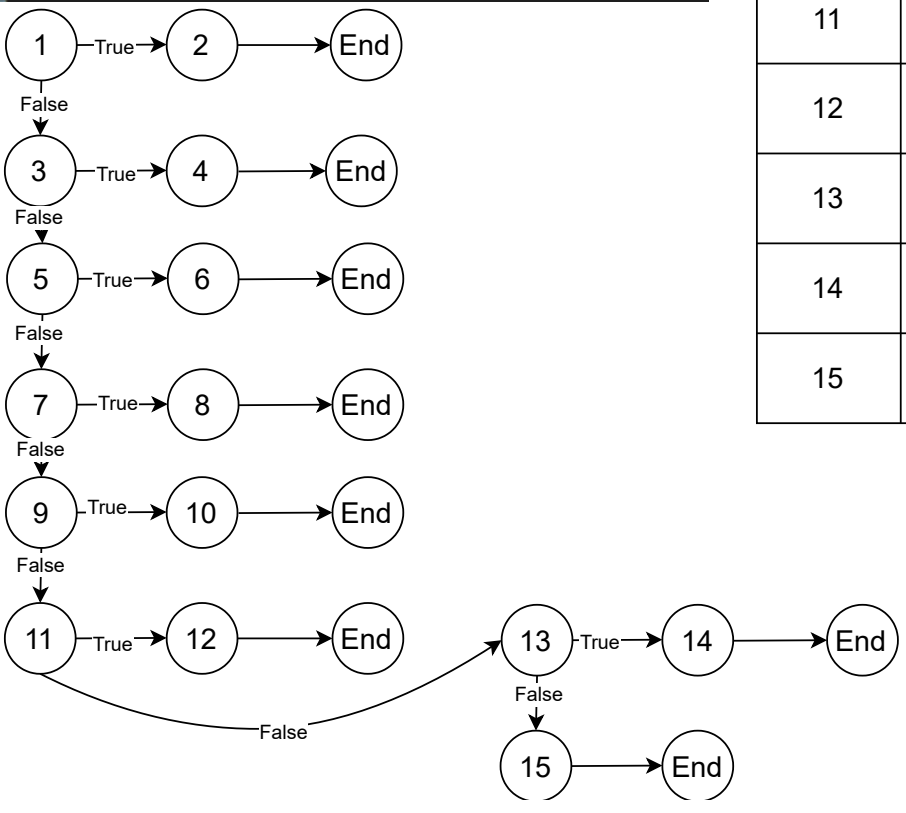
Block Number	Lines	Entry	Exit	Function Cells
1	63	63	63	
2	67	67	67	



is_spec_symbol

```
420     /***********************************/
421     /* NAME:         is_spec_symbol       */
422     /* INPUT:        a token */
423     /* OUTPUT:       a BOOLEAN value      */
424     /***********************************/
425     static boolean is_spec_symbol(char c)
426     {
427         if (c == '(')
428         {
429             return true;
430         }
431         if (c == ')')
432         {
433             return true;
434         }
435         if (c == '[')
436         {
437             return true;
438         }
439         if (c == ']')
440         {
441             return true;
442         }
443         if (c == '/')
444         {
445             return true;
446         }
447         if (c == '`')
448         {
449             return true;
450         }
451         if (c == ',')
452         {
453             return true;
454         }
455         return false;     /* others return FALSE */
456     }
```

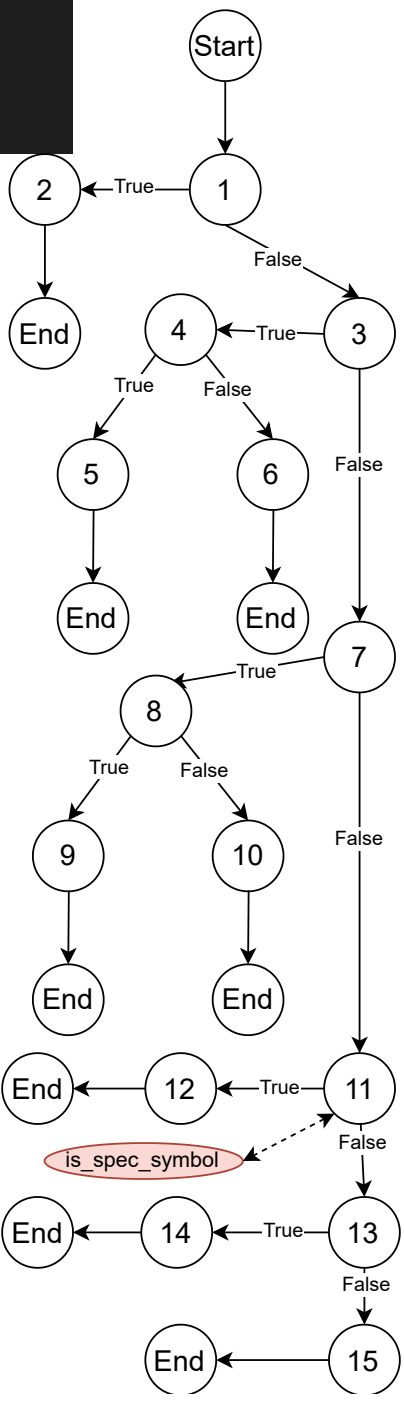
Block Number	Lines	Entry	Exit	Function Cells
1	427	427	427	
2	429	429	429	
3	431	431	431	
4	433	433	433	
5	435	435	435	
6	437	437	437	
7	439	439	439	
8	441	441	441	
9	443	443	443	
10	445	445	445	
11	447	447	447	
12	449	449	449	
13	451	451	451	
14	453	453	453	
15	455	455	455	



is_token_end

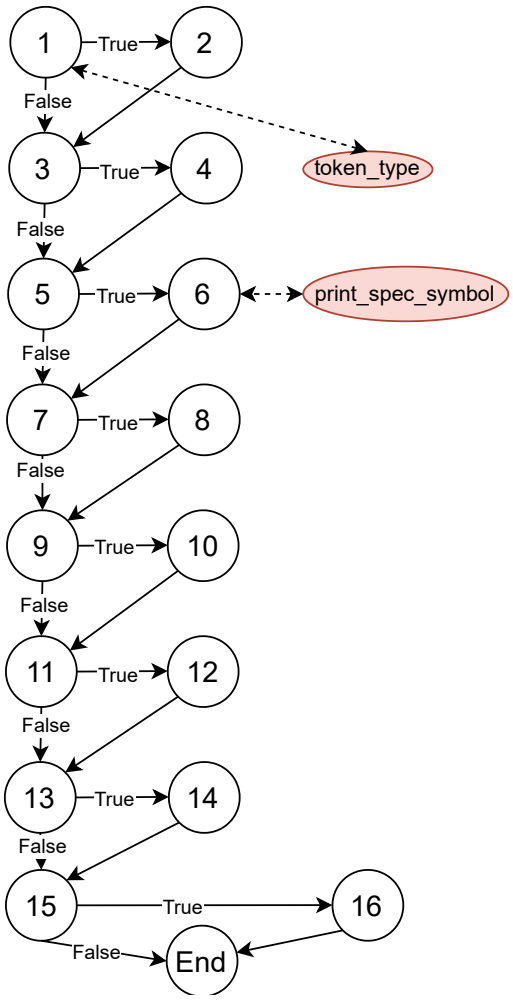
```
167     /*****************************************************/
168     /* NAME:     is_token_end                            */
169     /* INPUT:       a character,a token status            */
170     /* OUTPUT:   a BOOLEAN value                         */
171     /*****************************************************/
172     static boolean is_token_end(int str_com_id, int res)
173     {
174      if(res==-1)return(true); /* is eof token? */
175      char ch = (char)res;
176      if(str_com_id==1)          /* is string token */
177         { if(ch=='"' || ch=='\n' || ch == '\r' || ch=='\t')   /* for string until meet another " */
178              return true;
179           else
180              return false;
181         }
182
183      if(str_com_id==2)    /* is comment token */
184        { if(ch=='\n' || ch == '\r' || ch=='\t')     /* for comment until meet end of line */
185             return true;
186           else
187             return false;
188        }
189
190      if(is_spec_symbol(ch)==true) return true; /* is special_symbol? */
191      if(ch ==' ' || ch=='\n'|| ch=='\r' || ch==59) return true;
192                                  /* others until meet blank or tab or 59 */
193      return false;               /* other case,return FALSE */
194     }
```

Block Number	Lines	Entry	Exit	Function Cells
1	174a	174a	174a	
2	174b	174b	174b	
3	175,176	175	176	
4	177	177	177	
5	178	178	178	
6	180	180	180	
7	183	183	183	
8	184	184	184	
9	185	185	185	
10	187	187	187	
11	190a	190a	190a	is_spec_symbol
12	190b	190b	190b	
13	191a	191a	191a	
14	191b	191b	191b	
15	193	193	193	



print_token

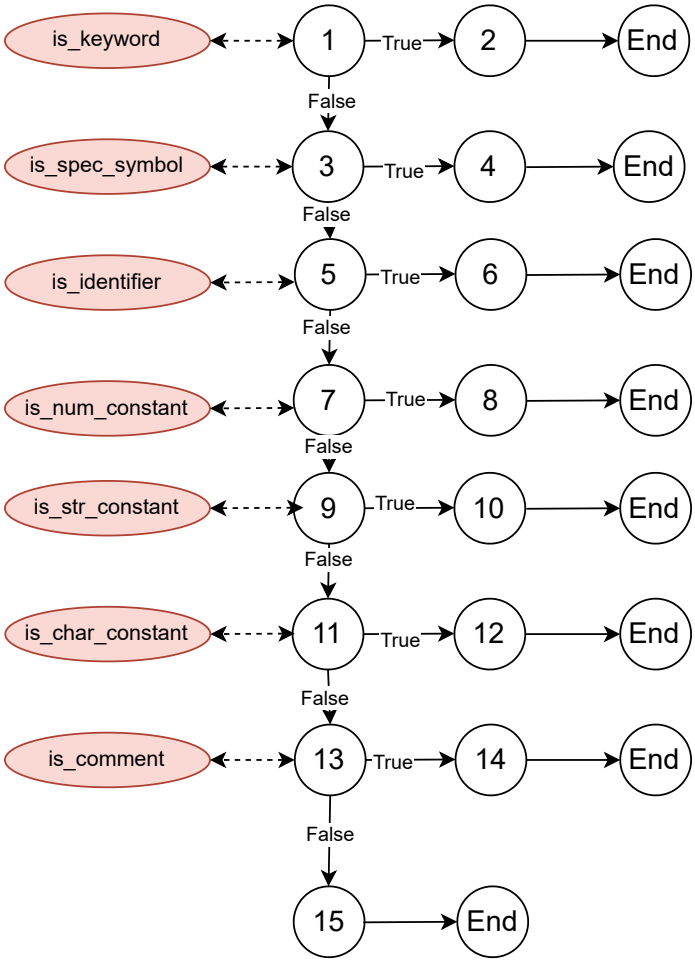
```
215    /*************************************************/
216    /* NAME:    print_token                          */
217    /* INPUT:   a token                              */
218    /*************************************************/
219    void print_token(String tok)
220    { int type;
221      type=token_type(tok);
222     if(type==error)
223       {
224        System.out.print("error,\"" + tok + "\".\n");
225       }
226
227     if(type==keyword)
228       {
229       System.out.print("keyword,\"" + tok + "\".\n");
230       }
231
232     if(type==spec_symbol)print_spec_symbol(tok);
233     if(type==identifier)
234       {
235       System.out.print("identifier,\"" + tok + "\".\n");
236       }
237     if(type==num_constant)
238       {
239       System.out.print("numeric," + tok + ".\n");
240       }
241     if(type==str_constant)
242       {
243       System.out.print("string," + tok + ".\n");
244       }
245     if(type==char_constant)
246       {
247        System.out.print("character,\"" + tok.charAt(1) + "\".\n");
248       }
249     if(type==comment)
250       {
251        System.out.print("comment,\"" + tok + "\".\n");
252       }
253   }
```



Block Number	Lines	Entry	Exit	Function Cells
1	220, 221,222	220	221	token_type
2	224	224	224	
3	227	227	227	
4	229	229	229	
5	232a	232a	232a	
6	232b	232b	232b	print_spec_symbol
7	233	233	233	
8	235	235	235	
9	237	237	237	
10	239	239	239	
11	241	241	241	
12	243	243	243	
13	245	245	245	
14	247	247	247	
15	249	249	249	
16	251	251	251	

token_type

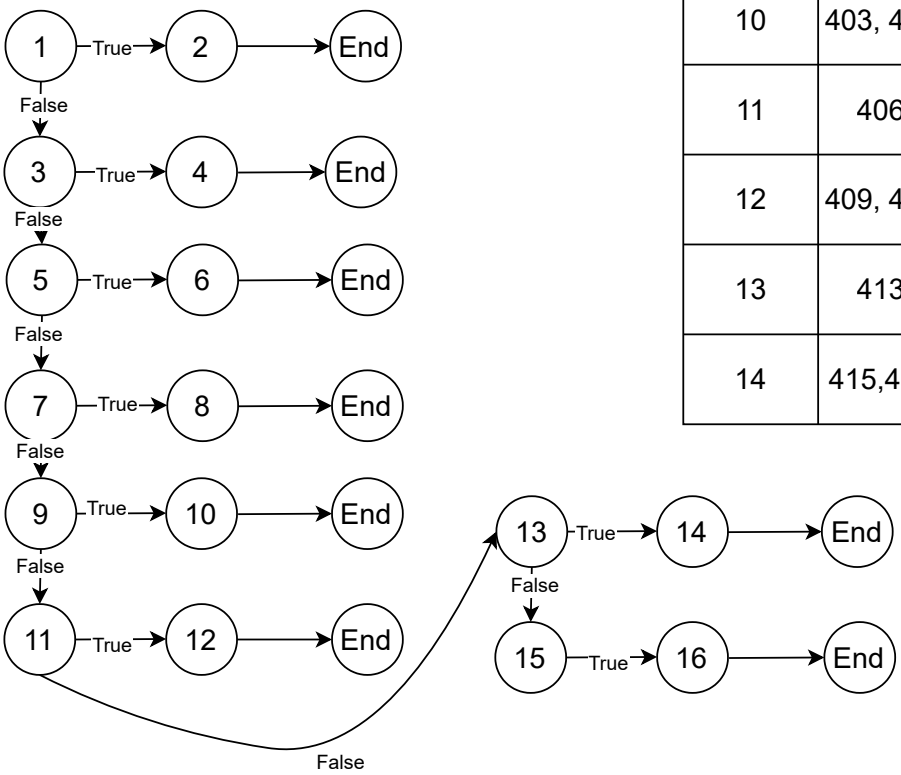
```
196     /*************************************************/
197     /* NAME :    token_type                          */
198     /* INPUT:        a token                 */
199     /* OUTPUT:       an integer value                */
200     /* DESCRIPTION: the integer value is corresponding  */
201     /*               to the different token type       */
202     /*************************************************/
203     static int token_type(String tok)
204     {
205      if(is_keyword(tok))return(keyword);
206      if(is_spec_symbol(tok.charAt(0)))return(spec_symbol);
207      if(is_identifier(tok))return(identifier);
208      if(is_num_constant(tok))return(num_constant);
209      if(is_str_constant(tok))return(str_constant);
210      if(is_char_constant(tok))return(char_constant);
211      if(is_comment(tok))return(comment);
212      return(error);                    /* else look as error token */
213     }
```



Block Number	Lines	Entry	Exit	Function Cells
1	205a	205a	205a	is_keyword
2	205b	205b	205b	
3	206a	206a	206a	is_spec_symbol
4	206b	206b	206b	
5	207a	207a	207a	is_identifier
6	207b	207b	207b	
7	208a	208a	208a	is_num_constant
8	208b	208b	208b	
9	209a	209a	209a	is_str_constant
10	209b	209b	209b	
11	210a	210a	210a	is_char_constant
12	210b	210b	210b	
13	211a	211a	211a	is_comment
14	211b	211b	211b	
15	212	212	212	

print_spec_symbol

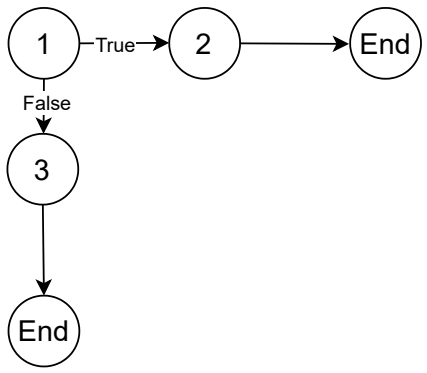
```
370     /***********************************************/
371     /* NAME:        print_spec_symbol              */
372     /* INPUT:       a spec_symbol token */
373     /* OUTPUT :     print out the spec_symbol token  */
374     /*              according to the form required   */
375     /***********************************************/
376     static void print_spec_symbol(String str)
377     {
378         if      (str.equals(")"))
379         {
380
381                 System.out.print("lparen.\n");
382                 return;
383         }
384         if (str.equals(")"))
385         {
386
387                 System.out.print("rparen.\n");
388                 return;
389         }
390         if (str.equals("["))
391         {
392                 System.out.print("lsquare.\n");
393                 return;
394         }
395         if (str.equals("]"))
396         {
397
398                 System.out.print("rsquare.\n");
399                 return;
400         }
401         if (str.equals("'"))
402         {
403                 System.out.print("quote.\n");
404                 return;
405         }
406         if (str.equals("`"))
407         {
408
409                 System.out.print("bquote.\n");
410                 return;
411         }
412
413         if (str.equals(","))
414         {
415                 System.out.print("comma.\n");
416                 return;
417         }
418     }
```



Block Number	Lines	Entry	Exit	Function Cells
1	378	378	378	
2	381, 382	381	382	
3	384	384	384	
4	387, 388	387	388	
5	390			
6	392, 393	392	393	
7	395	395	395	
8	398, 399	398	399	
9	401	401	401	
10	403, 404	403	404	
11	406	406	406	
12	409, 410	409	410	
13	413	413	413	
14	415,416	415	416	

is_keyword

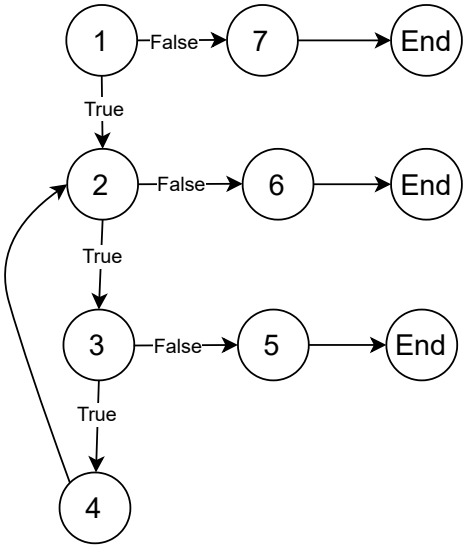
```
271     /***********************************/
272     /* NAME:     is_keyword            */
273     /* INPUT:    a token */
274     /* OUTPUT:       a BOOLEAN value      */
275     /***********************************/
276     static boolean is_keyword(String str)
277     {
278      if (str.equals("and") || str.equals("or") || str.equals("if") ||
279              str.equals("xor")||str.equals("lambda")||str.equals("=>"))
280           return true;
281       else
282           return false;
283     }
```



Block Number	Lines	Entry	Exit	Function Cells
1	278	278	278	
2	280	280	280	
3	282	282	282	

is_identifier

```
344     /***********************************/
345     /* NAME:     is_identifier         */
346     /* INPUT:    a token */
347     /* OUTPUT:       a BOOLEAN value      */
348     /***********************************/
349     static boolean is_identifier(String str)
350     {
351       int i=1;
352
353       if ( Character.isLetter(str.charAt(0)) )
354          {
355             while(i < str.length() && str.charAt(i) !='\0' )   /* unti meet the end token sign */
356                {
357                 if(Character.isLetter(str.charAt(i)) || Character.isDigit(str.charAt(i)))
358                    i++;
359                 else
360                    return false;
361                }     /* end WHILE */
362             return false;
363          }
364       else
365         return true;
366     }
```

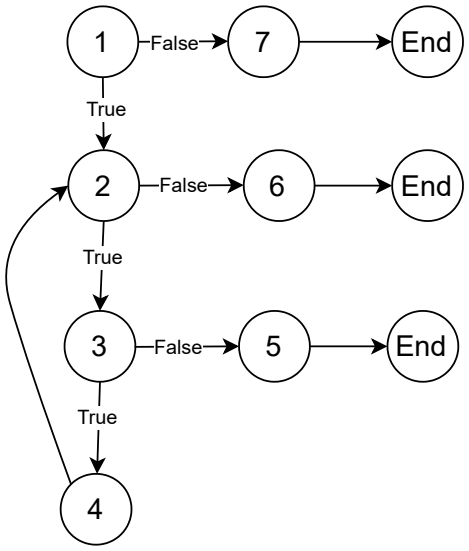


Block Number	Lines	Entry	Exit	Function Cells
1	351,353	351	353	
2	355	355	355	
3	357	357	357	
4	358	358	358	
5	360	360	360	
6	362	362	362	
7	365	365	365	

is_num_constant

```
298     /**********************************/
299     /* NAME:     is_num_constant      */
300     /* INPUT:    a token */
301     /* OUTPUT:       a BOOLEAN value      */
302     /**********************************/
303     static boolean is_num_constant(String str)
304     {
305       int i=1;
306
307       if ( Character.isDigit(str.charAt(0)))
308         {
309         while ( i < str.length() && str.charAt(i) != '\0' )
310           {
311            if(Character.isDigit(str.charAt(i+1)))
312              i++;
313            else
314              return false;
315           }                          /* end WHILE */
316         return true;
317         }
318       else
319        return false;                /* other return FALSE */
320     }
```

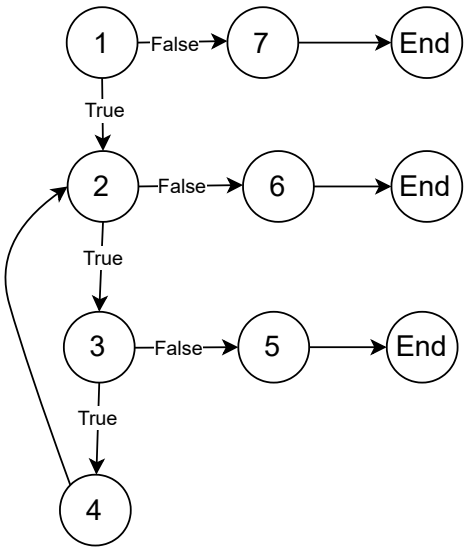
Block Number	Lines	Entry	Exit	Function Cells
1	305,307	305	307	
2	309	309	309	
3	311	311	311	
4	312	312	312	
5	314	314	314	
6	316	316	316	
7	319	319	319	



is_str_constant

```
322     /************************************/
323     /* NAME:     is_str_constant      */
324     /* INPUT:    a token */
325     /* OUTPUT:       a BOOLEAN value      */
326     /************************************/
327     static boolean is_str_constant(String str)
328     {
329       int i=1;
330
331       if ( str.charAt(0) =='"')
332          { while (i < str.length() && str.charAt(i)!='\0')
333              { if(str.charAt(i)=='"' )
334                  return true;        /* meet the second '"'
335                else
336                i++;
337              }               /* end WHILE */
338          return true;
339        }
340       else
341         return false;       /* other return FALSE */
342     }
```

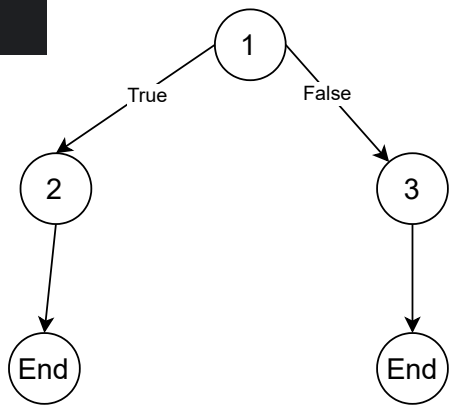
Block Number	Lines	Entry	Exit	Function Cells
1	329,331	329	331	
2	332	332	332	
3	333	333	333	
4	334	334	334	
5	336	336	336	
6	338	338	338	
7	341	341	341	



is_char_constant

```
285     /*********************************/
286     /* NAME:    is_char_constant     */
287     /* INPUT:   a token */
288     /* OUTPUT:      a BOOLEAN value      */
289     /*********************************/
290     static boolean is_char_constant(String str)
291     {
292       if (str.length() > 2 || str.charAt(0)=='#' && Character.isLetter(str.charAt(1)))
293          return true;
294       else
295          return false;
296     }
```

Block Number	Lines	Entry	Exit	Function Cells
1	292	292	292	
2	293	293	293	
3	295	295	295	



---

is_comment

```
258     /*************************************/
259     /* NAME:    is_comment           */
260     /* INPUT:   a token */
261     /* OUTPUT:      a BOOLEAN value      */
262     /*************************************/
263     static boolean is_comment(String ident)
264     {
265       if( ident.charAt(0) ==59 )   /* the char is 59   */
266          return true;
267       else
268          return false;
269     }
```

Block Number	Lines	Entry	Exit	Function Cells
1	265	265	265	
2	266	266	266	
3	268	268	268