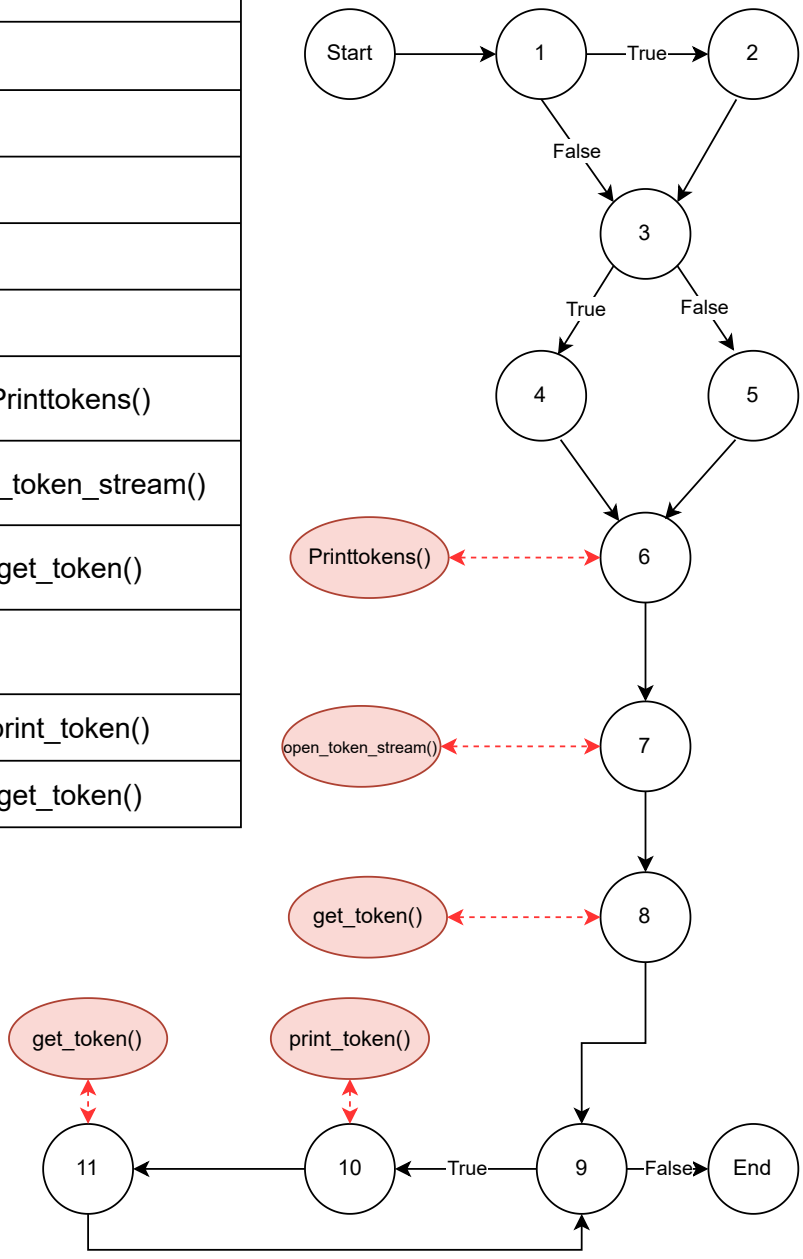
main

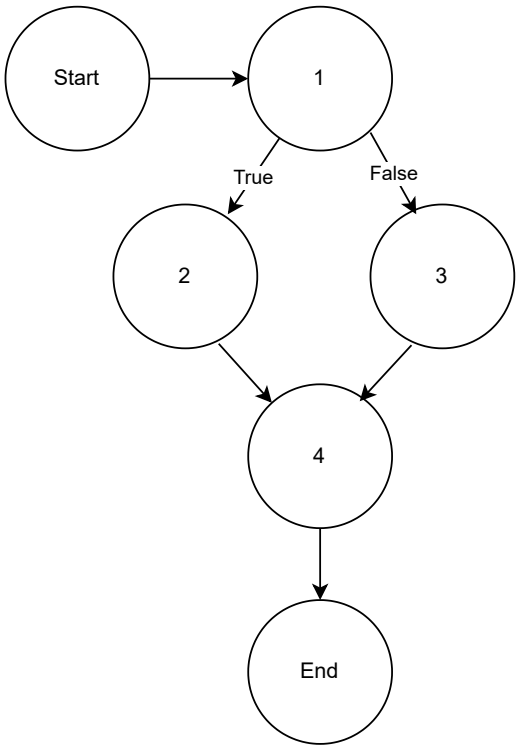
```
459     public static void main(String[] args)  {
460         String fname = null;
461         if (args.length == 0) { /* if not given filename,take as '""' */
462             fname = new String();
463         } else if (args.length == 1) {
464             fname = args[0];
465         } else {
466             System.out.print("Error! Please give the token stream\n");
467         }
468         Printtokens t = new Printtokens();
469         BufferedReader br = t.open_token_stream(fname); /* open token stream */
470         String tok = t.get_token(br);
471         while (tok != null) {   /* take one token each time until eof */
472             t.print_token(tok);
473             tok = t.get_token(br);
474 
475         }
476     }
```

Block Number	Lines	Entry	Exit	Function Calls
1	460, 461	460	461	
2	462	462	462	
3	463	463	463	
4	464	464	464	
5	466	466	466	
6	468	468	468	Printtokens()
7	469	469	469	open_token_stream()
8	470	470	470	get_token()
9	471	471	471	
10	472	472	472	print_token()
11	473	473	473	get_token()



open_character_stream

```
23      BufferedReader open_character_stream(String fname) {
24          BufferedReader br = null;
25          if (fname == null) {
26              br = new BufferedReader(new InputStreamReader(System.in));
27          } else {
28              try {
29                  FileReader fr = new FileReader(fname);
30                  br = new BufferedReader(fr);
31              } catch (FileNotFoundException e) {
32                  System.out.print("The file " + fname +" doesn't exists\n");
33                  e.printStackTrace();
34              }
35          }
36
37          return br;
38      }
```

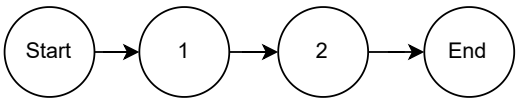


Block Number	Lines	Entry	Exit	Function Calls
1	24, 25	24	25	
2	26	26	26	
3	29, 30	29	30	
4	37	37	37	

get_char

```
45      int get_char(BufferedReader br){
46          int ch = 0;
47          try {
48              br.mark(4);
49              ch= br.read();
50          } catch (IOException e) {
51              e.printStackTrace();
52          }
53          return ch;
54      }
```

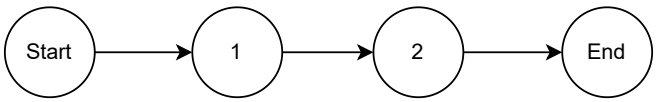
Block Number	Lines	Entry	Exit	Function Calls
1	46, 48, 49	46	49	
2	53	53	53	



unget_char

```
62      char unget_char (int ch,BufferedReader br) {
63        try {
64          br.reset();
65      } catch (IOException e) {
66          e.printStackTrace();
67      }
68           return 0;
69      }
```

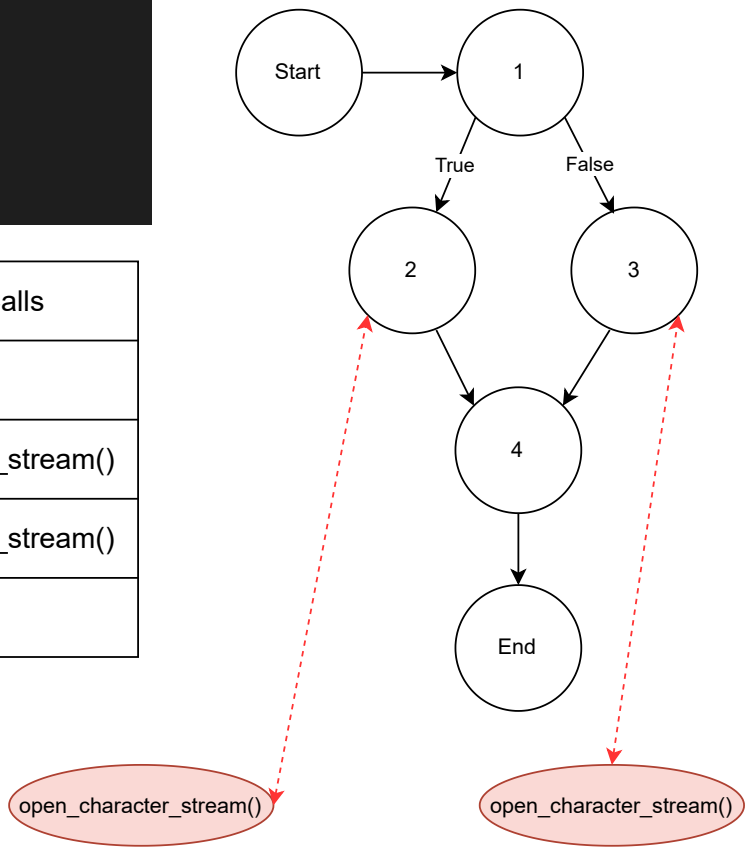
Block Number	Lines	Entry	Exit	Function Calls
1	64	64	64	
2	68	68	68	



open_token_stream

```
78      BufferedReader open_token_stream(String fname)
79      {
80          BufferedReader br;
81      if(fname==null || fname.equals(""))
82          br=open_character_stream(null);
83       else
84          br=open_character_stream(fname);
85       return br;
86      }
```

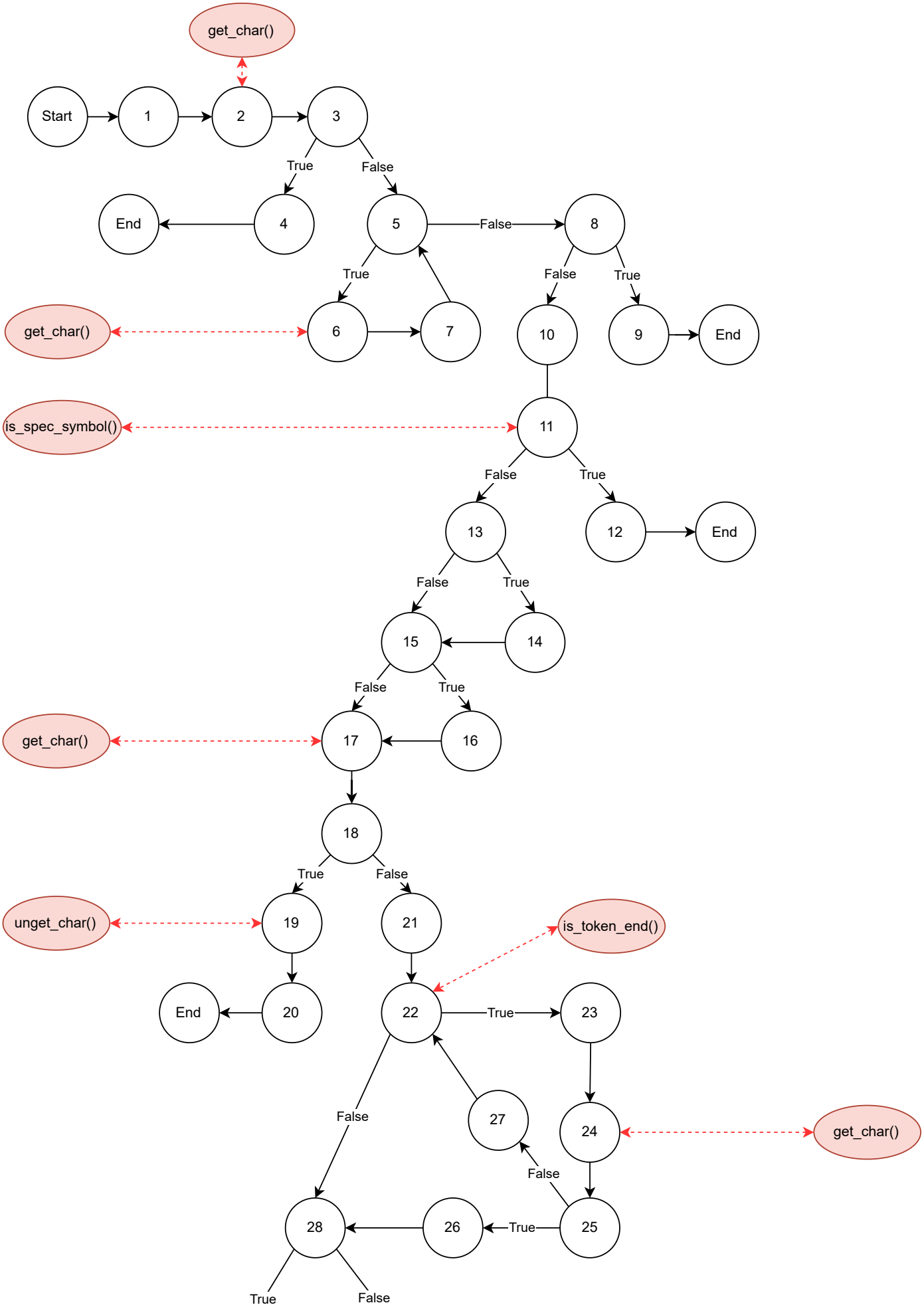
Block Number	Lines	Entry	Exit	Function Calls
1	80, 81	80	81	
2	82	82	82	open_character_stream()
3	84	84	84	open_character_stream()
4	85	85	85	

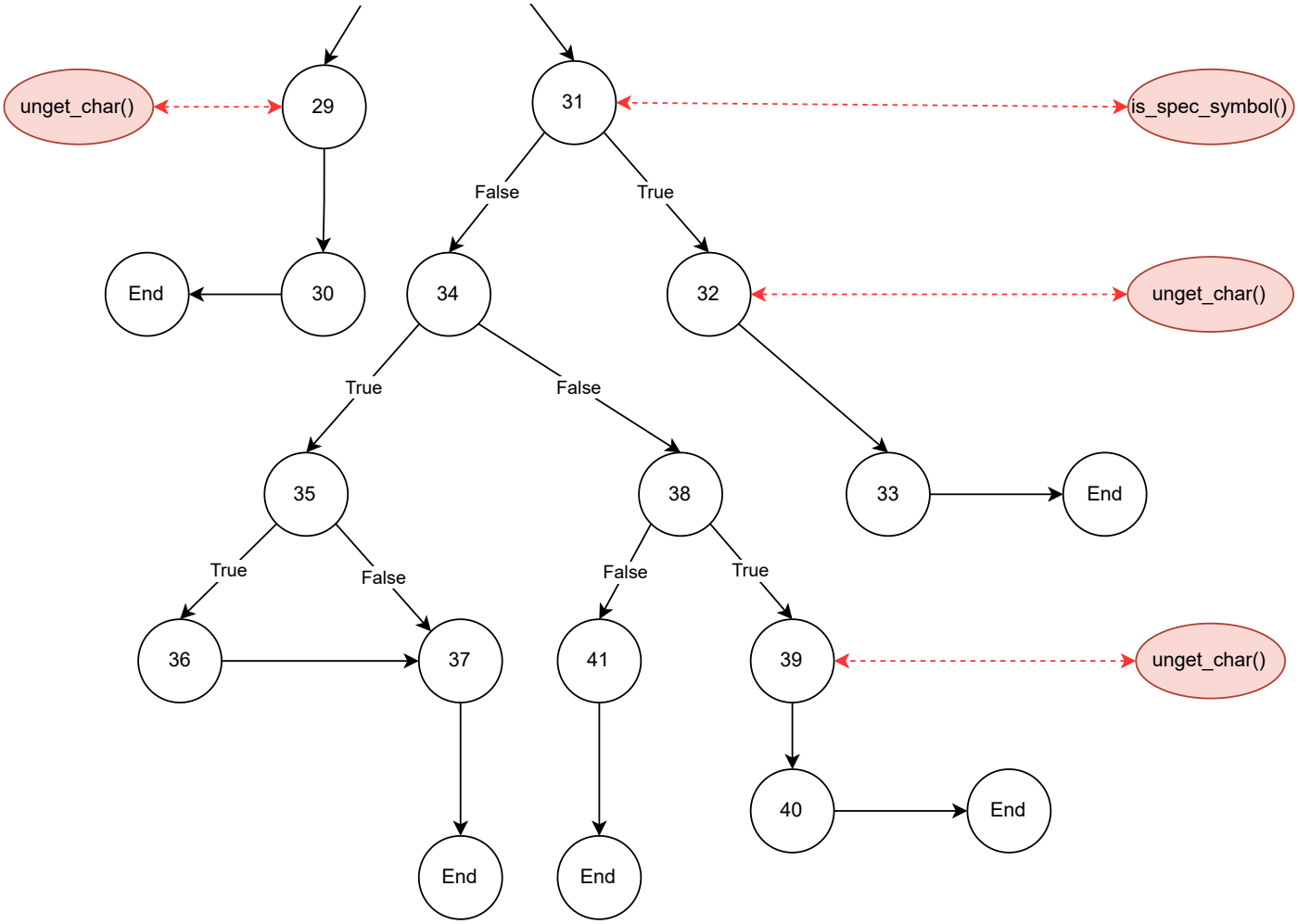


get_token

```
String get_token(BufferedReader br)
{
  int i=0,j;
  int id=0;
  int res = 0;
  char ch = '\0';

  StringBuilder sb = new StringBuilder();

   try {
       res = get_char(br);
       if (res == -1) {
           return null;
       }
       ch = (char)res;
      while(ch==' '||ch=='\n' || ch == '\r')
      {
        res = get_char(br);
        ch = (char)res;
      }

   if(res == -1)return null;
   sb.append(ch);
   if(is_spec_symbol(ch)==true)return sb.toString();
   if(ch =='"')id=2;    /* prepare for string */
   if(ch ==59)id=1;    /* prepare for comment */

   res = get_char(br);
   if (res == -1) {
       unget_char(ch,br);
       return sb.toString();
   }
   ch = (char)res;

   while (is_token_end(id,res) == false)/*
   {
       sb.append(ch);
       br.mark(4);
       res = get_char(br);
       if (res == -1) {
           break;
       }
       ch = (char)res;
   }

   if(res == -1)       /* if end character
      { unget_char(ch,br);        /* then
        return sb.toString();
      }

   if(is_spec_symbol(ch)==true)     /* if
      { unget_char(ch,br);        /* then
        return sb.toString();
      }
   if(id==1)                   /* if end ch
     {
       if (ch == '"') {
           sb.append(ch);
       }
       return sb.toString();
     }
    if(id==0 && ch==59)

      { unget_char(ch,br);
        return sb.toString();
      }
  } catch (IOException e) {
     e.printStackTrace();
  }

   return sb.toString();
  }
```

Block	Lines	Entry	Exit	Function Calls
1	97, 98, 99, 100, 102	97	102	
2	105	105	105	get_char()
3	106	106	106	
4	107	107	107	
5	109, 110	109	110	
6	112	112	112	get_char()
7	113	113	113	
8	116a	116a	116a	
9	116b	116b	116b	
10	117	117	117	
11	118a	118a	118a	is_spec_symbol()
12	118b	118b	118b	
13	119a	119a	119a	
14	119b	119b	119b	
15	120a	120a	120a	
16	120b	120b	120b	
17	122	122	122	get_char()
18	123	123	123	
19	124	124	124	unget_char()
20	125	125	125	
21	127	127	127	
22	129	129	129	is_token_end()
23	131, 132	131	132	
24	133	133	133	get_char()
25	134	134	134	
26	135	135	135	
27	137	137	137	
28	140	140	140	
29	141	141	141	unget_char()
30	142	142	142	
31	145	145	145	is_spec_symbol()
32	146	146	146	unget_char()
33	147	147	147	
34	149	149	149	
35	151	151	151	
36	152	152	152	
37	154	154	154	
38	156	156	156	
39	158	158	158	unget_char()
40	159	159	159	
41	165	165	165	

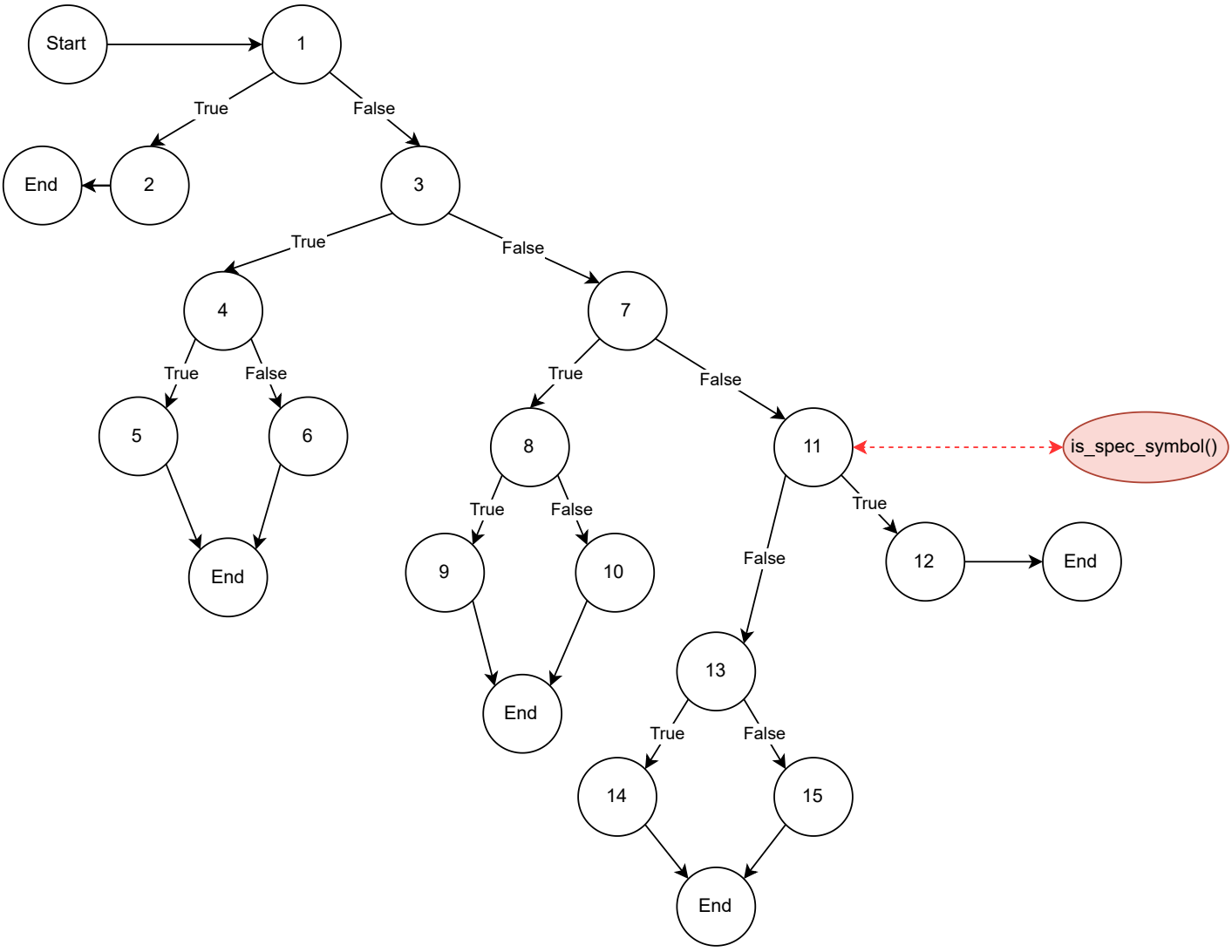

get_char()
Start
1
2
3
True
False
End
4
5
False
8
True
False
True
get_char()
6
7
10
9
End
is_spec_symbol()
11
False
True
13
12
End
False
True
15
14
False
True
get_char()
17
16
18
True
False
unget_char()
19
21
is_token_end()
End
20
22
True
23
False
27
24
get_char()
False
28
26
True
25
True
False



is_token_end

```
173     static boolean is_token_end(int str_com_id, int res)
174     {
175      if(res==-1)return(true); /* is eof token? */
176      char ch = (char)res;
177      if(str_com_id==1)          /* is string token */
178         { if(ch=='"' || ch=='\n' || ch == '\r' || ch=='\t')   /* for string until meet another " */
179              return true;
180           else
181              return false;
182         }
183
184      if(str_com_id==2)    /* is comment token */
185        { if(ch=='\n' || ch == '\r' || ch=='\t')     /* for comment until meet end of line */
186             return true;
187           else
188             return false;
189        }
190
191      if(is_spec_symbol(ch)==true) return true; /* is special_symbol? */
192      if(ch ==' ' || ch=='\n'|| ch=='\r' || ch==59) return true;
193                                  /* others until meet blank or tab or 59 */
194      return false;               /* other case,return FALSE */
195     }
```

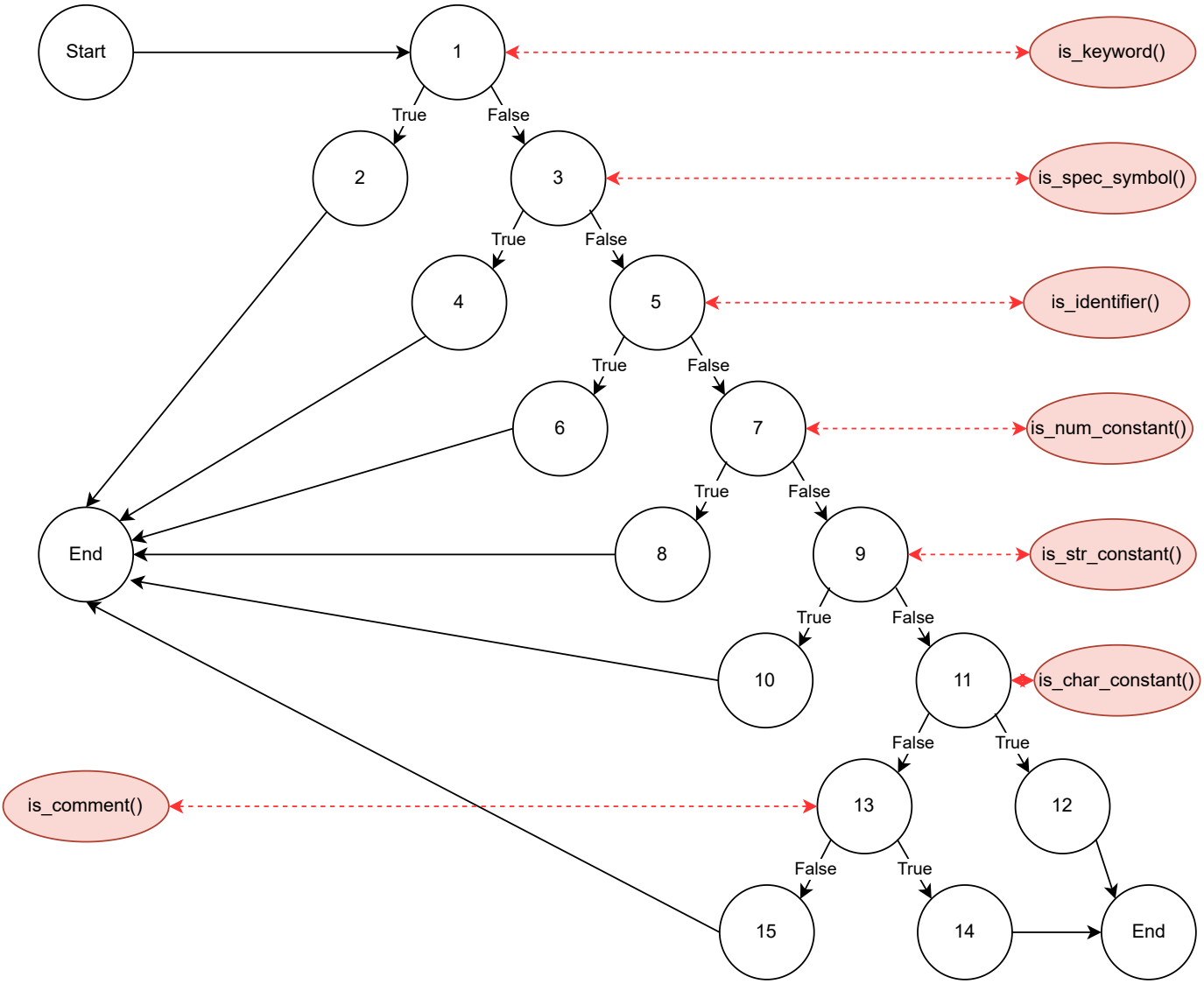
Block Number	Lines	Entry	Exit	Function Calls
1	175a	175a	175a	
2	175b	175b	175b	
3	176, 177	176	177	
4	178	178	178	
5	179	179	179	
6	181	181	181	
7	184	184	184	
8	185	185	185	
9	186	186	186	
10	188	188	188	
11	191a	191a	191a	is_spec_symbol()
12	191b	191b	191b	
13	192a	192a	192a	
14	192b	192b	192b	
15	194	194	194	



token_type

```
204     static int token_type(String tok)
205     {
206      if(is_keyword(tok))return(keyword);
207      if(is_spec_symbol(tok.charAt(0)))return(spec_symbol);
208      if(is_identifier(tok))return(identifier);
209      if(is_num_constant(tok))return(num_constant);
210      if(is_str_constant(tok))return(str_constant);
211      if(is_char_constant(tok))return(char_constant);
212      if(is_comment(tok))return(comment);
213      return(error);                    /* else look as error token */
214     }
```

Block Number	Lines	Entry	Exit	Function Calls
1	206a	206a	206a	is_keyword()
2	206b	206b	206b	
3	207a	207a	207a	is_spec_symbol()
4	207b	207b	207b	
5	208a	208a	208a	is_identifier()
6	208b	208b	208b	
7	209a	209a	209a	is_num_constant()
8	209b	209b	209b	
9	210a	210a	210a	is_str_constant()
10	210b	210b	210b	
11	211a	211a	211a	is_char_constant()
12	211b	211b	211b	
13	212a	212a	212a	is_comment()
14	212b	212b	212b	
15	213	213	213	

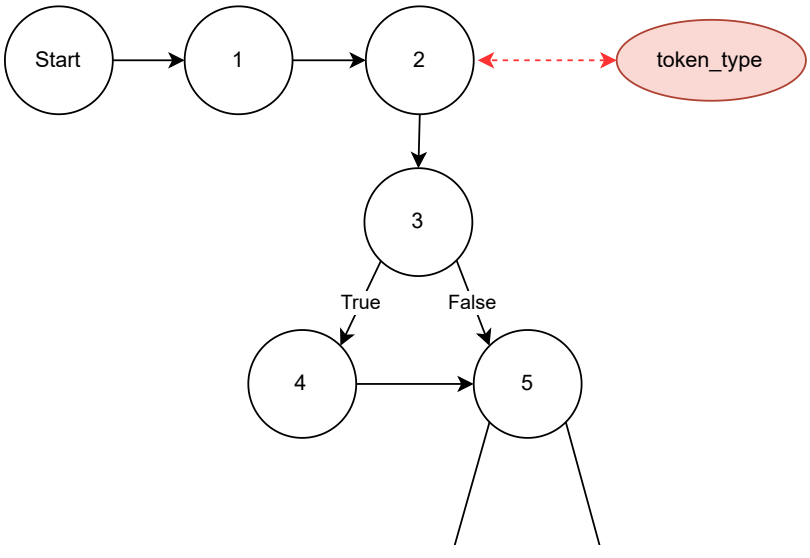


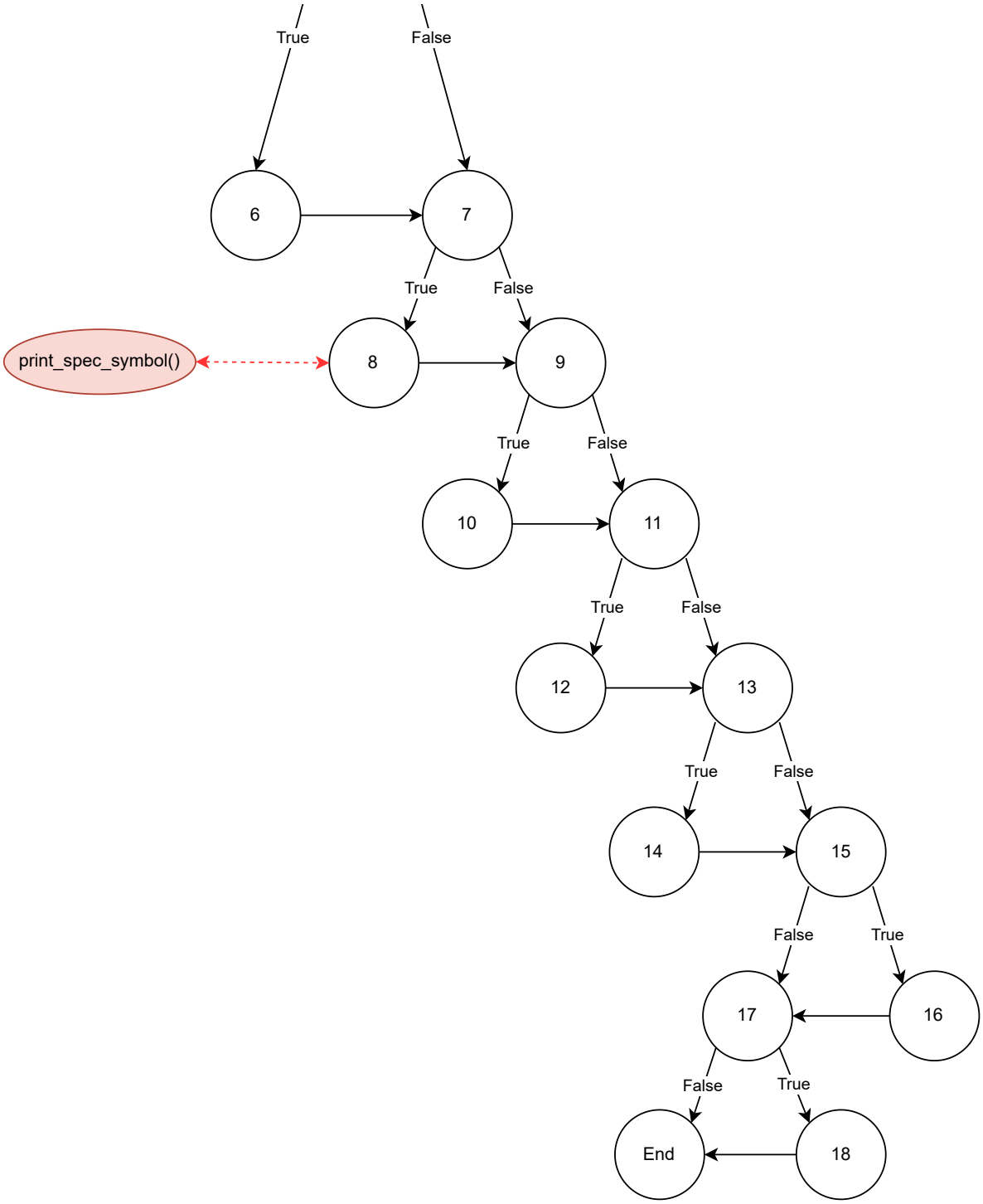
print_token

```
220     void print_token(String tok)
221     { int type;
222       type=token_type(tok);
223      if(type==error)
224        {
225         System.out.print("error,\"" + tok + "\".\n");
226        }
227
228      if(type==keyword)
229        {
230        System.out.print("keyword,\"" + tok + "\".\n");
231        }
232
233      if(type==spec_symbol)print_spec_symbol(tok);
234      if(type==identifier)
235        {
236        System.out.print("identifier,\"" + tok + "\".\n");
237        }
238      if(type==num_constant)
239        {
240        System.out.print("numeric," + tok + ".\n");
241        }
242      if(type==str_constant)
```

```
243         {
244           System.out.print("string," + tok + ".\n");
245           }
246         if(type==char_constant)
247           {
248            System.out.print("character,\"" + tok.charAt(1) + "\".\n");
249           }
250         if(type==comment)
251           {
252            System.out.print("comment,\"" + tok + "\".\n");
253           }
254       }
```

Block Number	Lines	Entry	Exit	Function Calls
1	221	221	221	
2	222	222	222	token_type()
3	223	223	223	
4	225	225	225	
5	228	228	228	
6	230	230	230	
7	233a	233a	233a	
8	233b	233b	233b	print_spec_symbol()
9	234	234	234	
10	236	236	236	
11	238	238	238	
12	240	240	240	
13	242	242	242	
14	244	244	244	
15	246	246	246	
16	248	248	248	
17	250	250	250	
18	252	252	252	

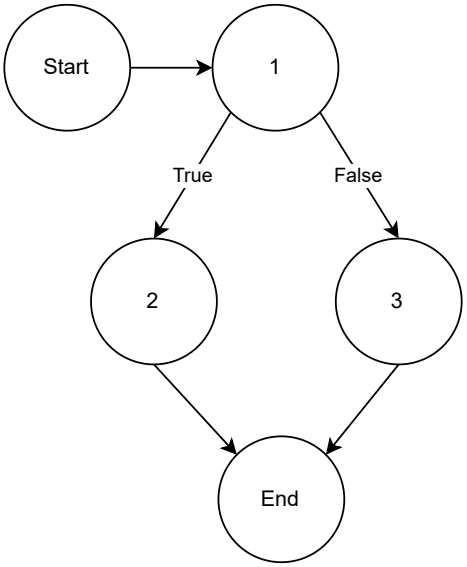




is_comment

```
264     static boolean is_comment(String ident)
265     {
266       if( ident.charAt(0) ==59 )   /* the char is 59   */
267          return true;
268       else
269          return false;
270     }
```

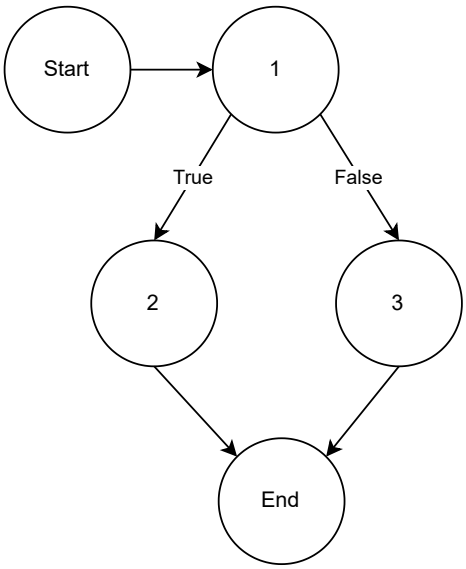
Block Number	Lines	Entry	Exit	Function Calls
1	266	266	266	
2	267	267	267	
3	269	269	269	



is_keyword

```
277     static boolean is_keyword(String str)
278     {
279      if (str.equals("and") || str.equals("or") || str.equals("if") ||
280              str.equals("xor")||str.equals("lambda")||str.equals("=>"))
281          return true;
282       else
283          return false;
284     }
```

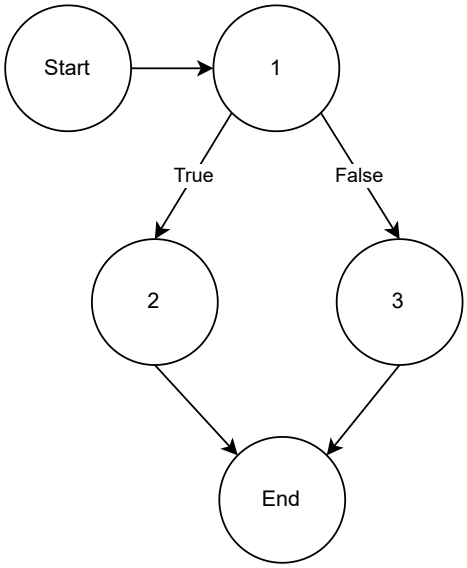
Block Number	Lines	Entry	Exit	Function Calls
1	279, 280	279	280	
2	281	281	281	
3	283	283	283	



is_char_constant

```
291     static boolean is_char_constant(String str)
292     {
293       if (str.length() > 2 || str.charAt(0)=='#' && Character.isLetter(str.charAt(1)))
294          return true;
295       else
296          return false;
297     }
```

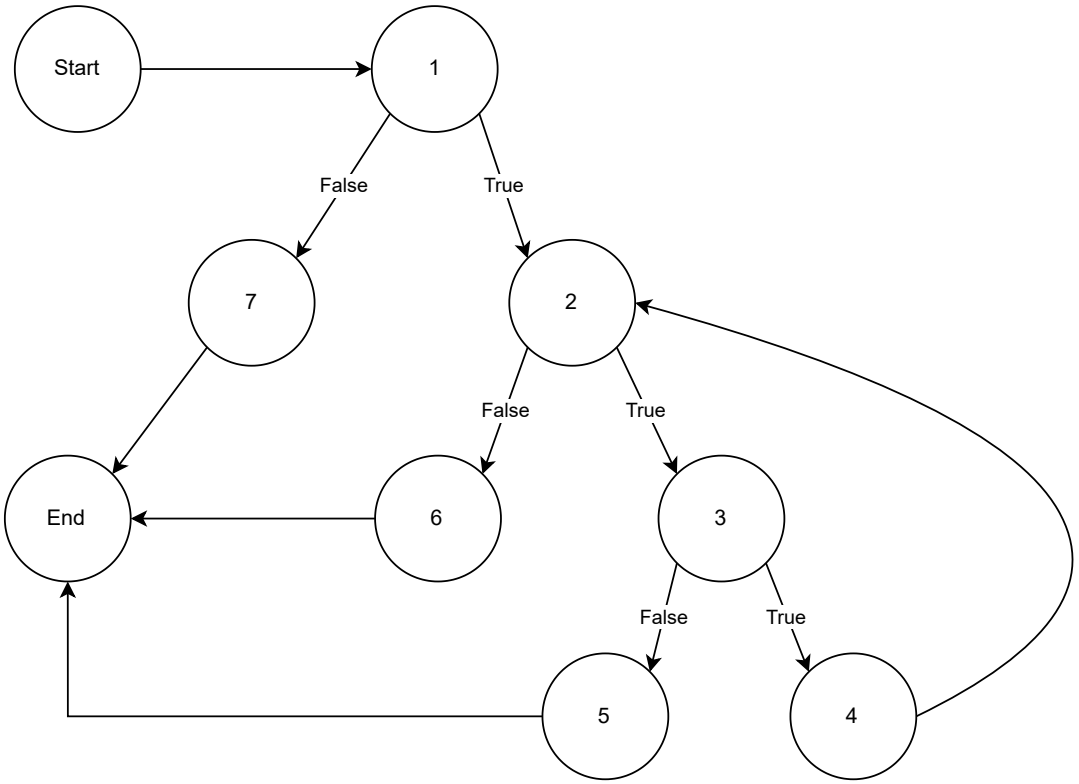
Block Number	Lines	Entry	Exit	Function Calls
1	293	293	293	
2	294	294	294	
3	296	296	296	



is_num_constant

```
304     static boolean is_num_constant(String str)
305     {
306       int i=1;
307
308       if ( Character.isDigit(str.charAt(0)))
309         {
310         while ( i < str.length() && str.charAt(i) != '\0' )
311           {
312            if(Character.isDigit(str.charAt(i+1)))
313              i++;
314            else
315              return false;
316           }                          /* end WHILE */
317         return true;
318         }
319       else
320        return false;               /* other return FALSE */
321     }
```

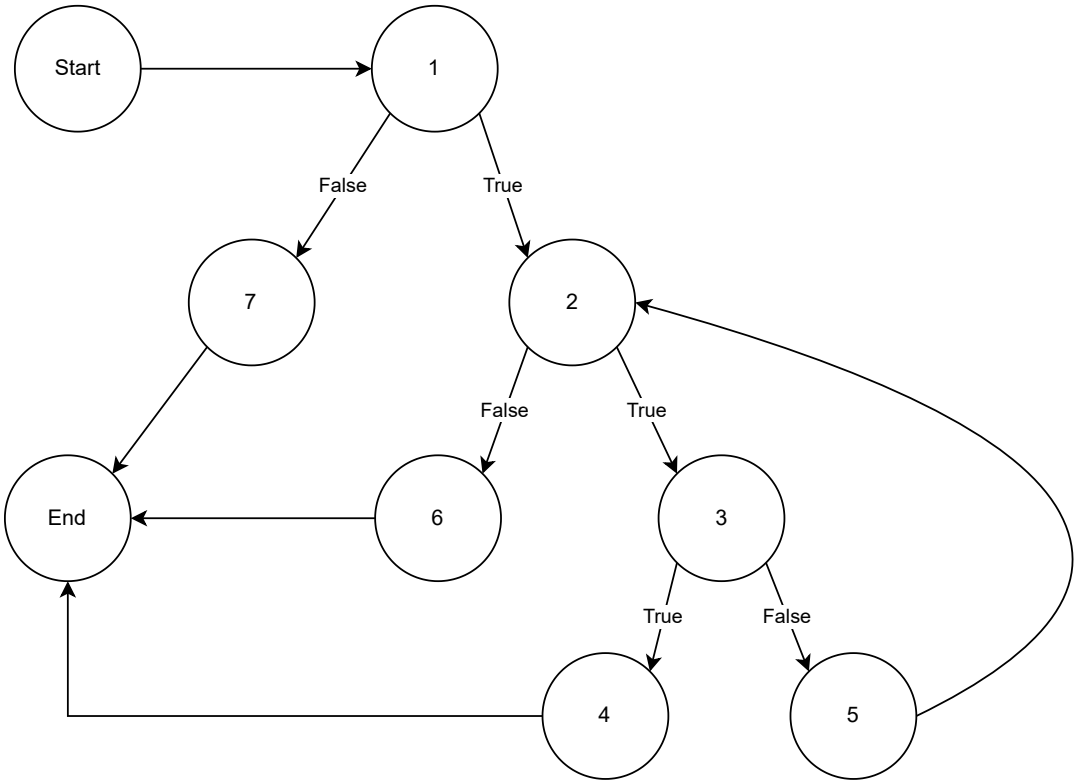
Block Number	Lines	Entry	Exit	Function Calls
1	306, 308	306	308	
2	310	310	310	
3	312	312	312	
4	313	313	313	
5	315	315	315	
6	317	317	317	
7	320	320	320	



is_str_constant

```
328     static boolean is_str_constant(String str)
329     {
330       int i=1;
331
332       if ( str.charAt(0) =='"')
333          { while (i < str.length() && str.charAt(i)!='\0')
334              { if(str.charAt(i)=='"' )
335                  return true;        /* meet the second '"'          */
336                else
337                i++;
338              }               /* end WHILE */
339           return true;
340         }
341       else
342         return false;       /* other return FALSE */
343     }
```

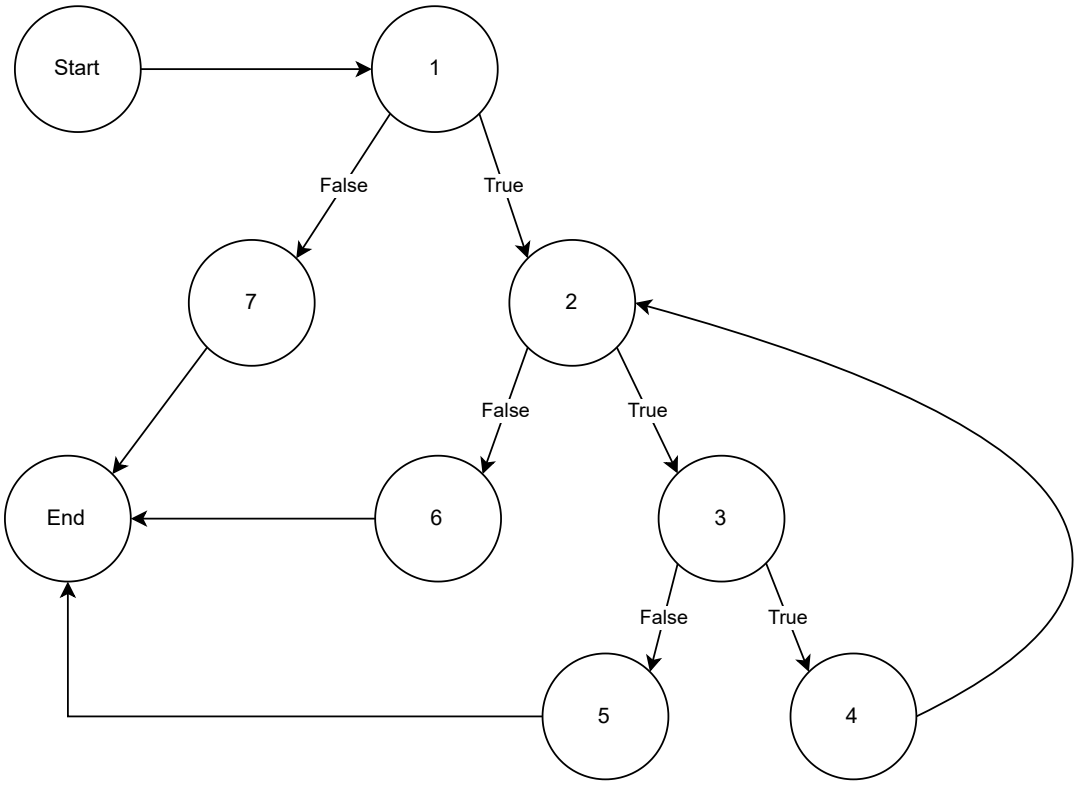
Block Number	Lines	Entry	Exit	Function Calls
1	330, 332	330	332	
2	333	333	333	
3	334	334	334	
4	335	335	335	
5	337	337	337	
6	339	339	339	
7	342	342	342	



is_identifier

```
350     static boolean is_identifier(String str)
351     {
352       int i=1;
353
354       if ( Character.isLetter(str.charAt(0)) )
355          {
356             while(i < str.length() && str.charAt(i) !='\0' )   /* unti meet the end token sign */
357                {
358                 if(Character.isLetter(str.charAt(i)) || Character.isDigit(str.charAt(i)))
359                    i++;
360                 else
361                    return false;
362                }      /* end WHILE */
363             return false;
364          }
365       else
366         return true;
367     }
```

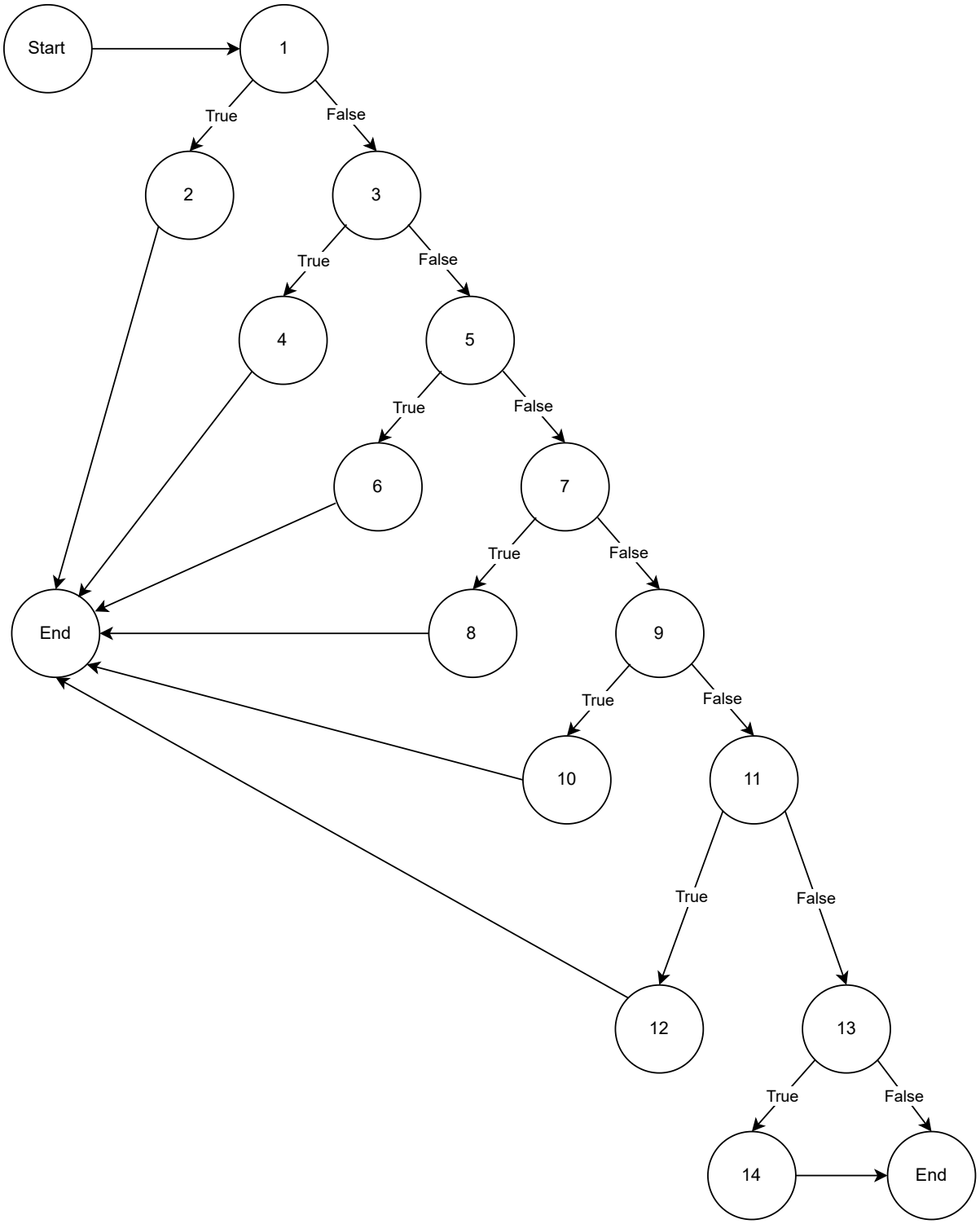
Block Number	Lines	Entry	Exit	Function Calls
1	352, 354	352	354	
2	356	356	356	
3	358	358	358	
4	359	359	359	
5	361	361	361	
6	363	363	363	
7	366	366	366	



print_spec_symbol

```
377     static void print_spec_symbol(String str)
378     {
379         if      (str.equals(")"))
380         {
381
382                 System.out.print("lparen.\n");
383                 return;
384         }
385         if (str.equals(")"))
386         {
387
388                 System.out.print("rparen.\n");
389                 return;
390         }
391         if (str.equals("["))
392         {
393                 System.out.print("lsquare.\n");
394                 return;
395         }
396         if (str.equals("]"))
397         {
398
399                 System.out.print("rsquare.\n");
400                 return;
401         }
402         if (str.equals("'"))
403         {
404                 System.out.print("quote.\n");
405                 return;
406         }
407         if (str.equals("`"))
408         {
409
410                 System.out.print("bquote.\n");
411                 return;
412         }
413
414         if (str.equals(","))
415         {
416                 System.out.print("comma.\n");
417                 return;
418         }
419     }
```

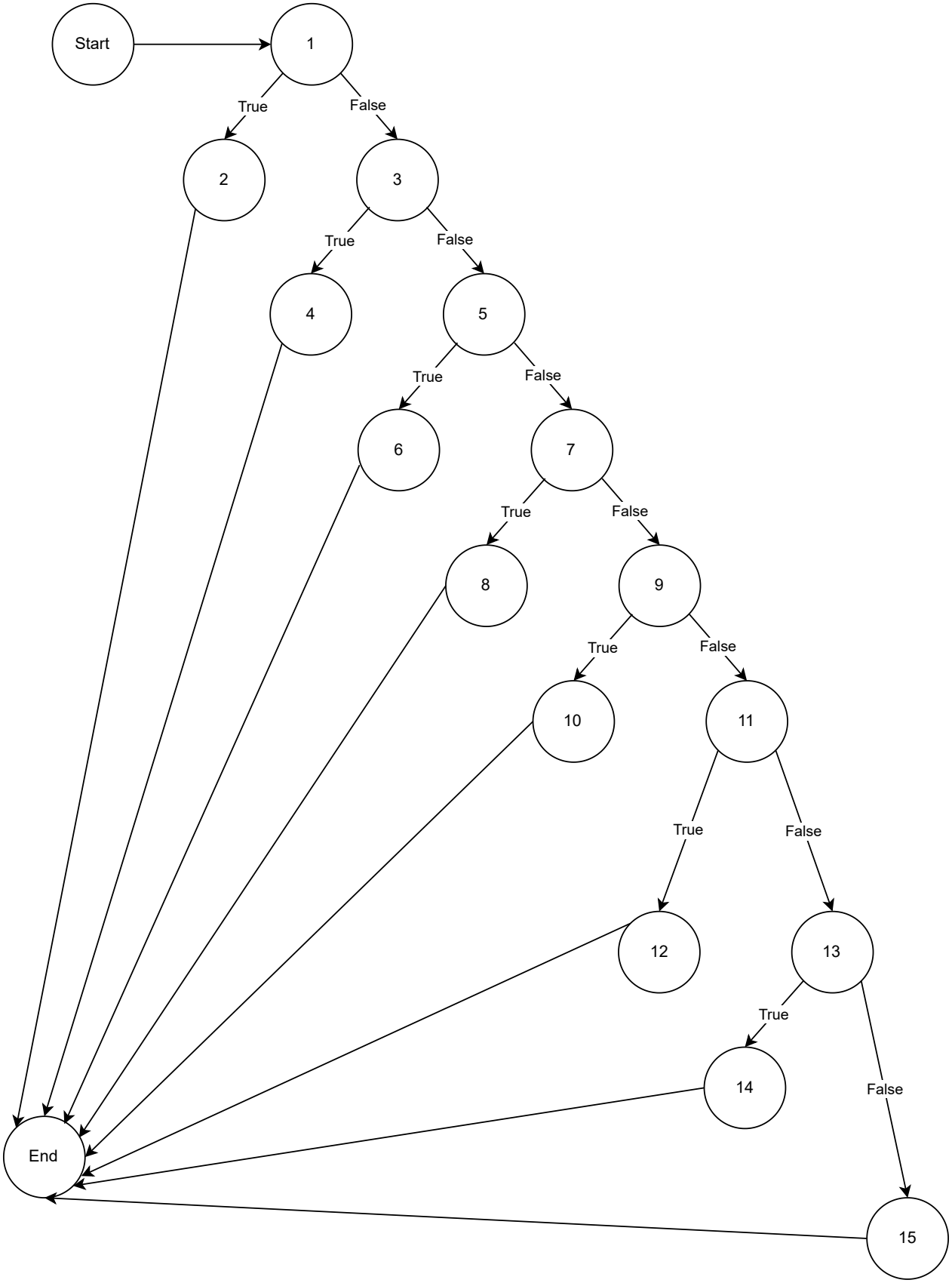
Block Number	Lines	Entry	Exit	Function Calls
1	379	379	379	
2	382, 383	382	383	
3	385	385	385	
4	388, 389	388	389	
5	391	391	391	
6	393, 394	393	394	
7	396	396	396	
8	399, 400	399	400	
9	402	402	402	
10	404, 405	404	405	
11	407	407	407	
12	410, 411	410	411	
13	414	414	414	
14	416, 417	416	417	


Start
1
True
False
2
3
True
False
4
5
True
False
6
7
True
False
End
8
9
True
False
10
11
True
False
12
13
True
False
14
End

is_spec_symbol

```
426     static boolean is_spec_symbol(char c)
427     {
428         if (c == '(')
429         {
430             return true;
431         }
432         if (c == ')')
433         {
434             return true;
435         }
436         if (c == '[')
437         {
438             return true;
439         }
440         if (c == ']')
441         {
442             return true;
443         }
444         if (c == '/')
445         {
446             return true;
447         }
448         if (c == '`')
449         {
450             return true;
451         }
452         if (c == ',')
453         {
454             return true;
455         }
456         return false;     /* others return FALSE */
457     }
```

Block Number	Lines	Entry	Exit	Function Calls
1	428	428	428	
2	430	430	430	
3	432	432	432	
4	434	434	434	
5	436	436	436	
6	438	438	438	
7	440	440	440	
8	442	442	442	
9	444	444	444	
10	446	446	446	
11	448	448	448	
12	450	450	450	
13	452	452	452	
14	454	454	454	
15	456	456	456	


Start
1
True
False
2
3
True
False
4
5
True
False
6
7
True
False
8
9
True
False
10
11
True
False
12
13
True
False
14
15
End