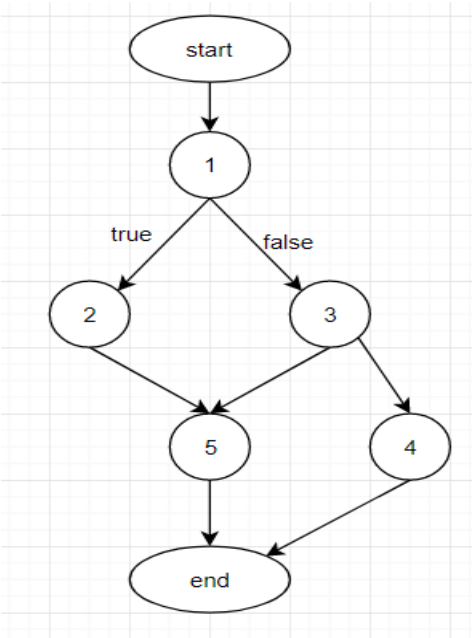
CFG:

1) **open_character_stream()**

```
28    BufferedReader open_character_stream(String fname) {
29        BufferedReader br = null;
30        if (fname == null) {
31            br = new BufferedReader(new InputStreamReader(System.in));
32        } else {
33            try {
34                FileReader fr = new FileReader(fname);
35                br = new BufferedReader(fr);
36            } catch (FileNotFoundException e) {
37                System.out.print("The file " + fname +" doesn't exists\n");
38                e.printStackTrace();
39            }
40        }
41
42        return br;
43    }
```

Block	Lines	Entry	Exit
1	29,30	29	30
2	31	31	31
3	34,35	34	35
4	37,38	37	38
5	42	42	42



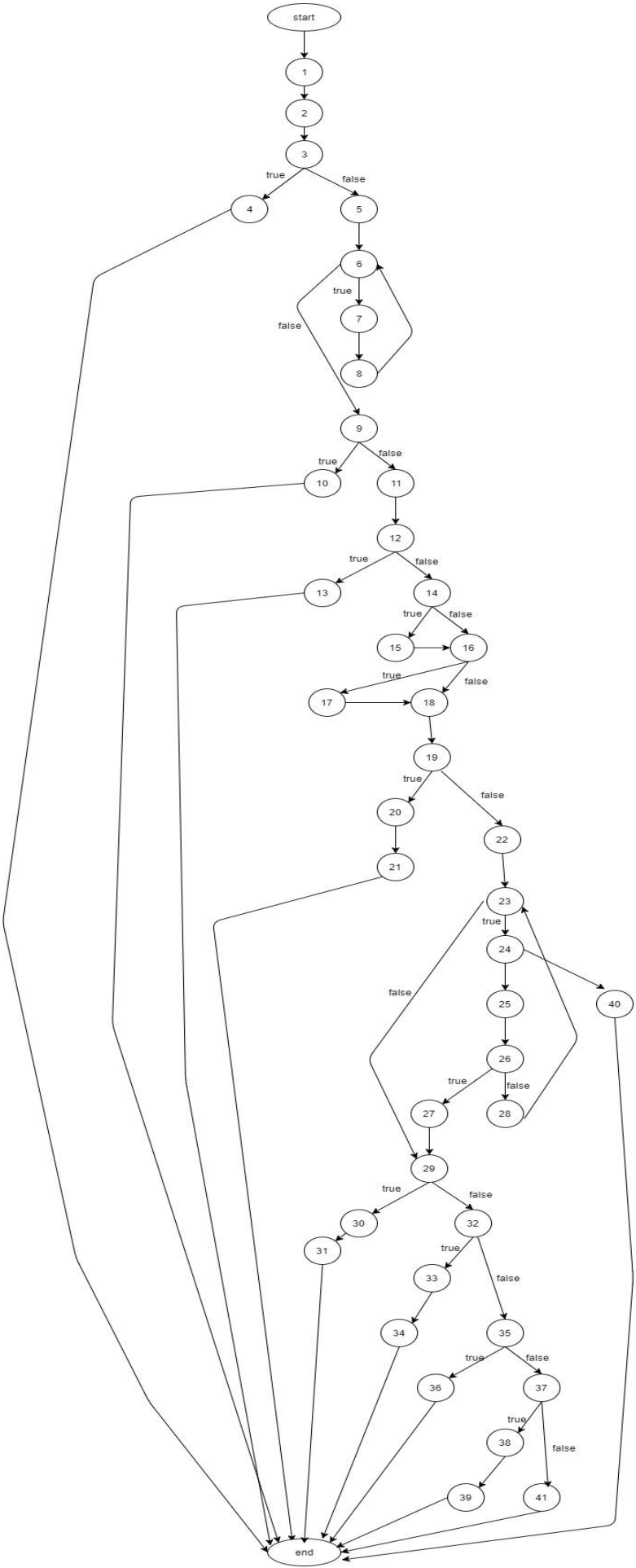
Edge Coverage:

1) [1,2,5]: fname=null [correction: fname=“”]
2) [1,3,5]: fname=”input.txt”
3) [1,3,4]: fname=”something”

2) get_token()

```
100    String get_token(BufferedReader br)
101    {
102      int i=0,j;
103      int id=0;
104      int res = 0;
105      char ch = '\0';
106
107      StringBuilder sb = new StringBuilder();
108
109       try {
110           res = get_char(br);
111           if (res == -1) {
112               return null;
113           }
114           ch = (char)res;
115        while(ch==' '||ch=='\n' || ch == '\r')      /* strip all blanks until meet characters */
116          {
117            res = get_char(br);
118            ch = (char)res;
119          }
120
121       if(res == -1)
122           return null;
123       sb.append(ch);
124       if(is_spec_symbol(ch)==true)
125           return sb.toString();
126       if(ch =='"')
127           id=0;    /* prepare for string */
128       if(ch ==59)
129           id=1;    /* prepare for comment */
130
131       res = get_char(br);
132       if (res == -1) {
133           unget_char(ch,br);
134           return sb.toString();
```

```
135         }
136         ch = (char)res;
137 
138         while (is_token_end(id,res) == false)/* until meet the end character */
139         {
140             sb.append(ch);
141             br.mark(4);
142             res = get_char(br);
143             if (res == -1) {
144                 break;
145             }
146             ch = (char)res;
147         }
148 
149         if(res == -1)       /* if end character is eof token    */
150            { unget_char(ch,br);        /* then put back eof on token_stream */
151              return sb.toString();
152            }
153 
154         if(is_spec_symbol(ch)==true)     /* if end character is special_symbol */
155            { unget_char(ch,br);        /* then put back this character       */
156              return sb.toString();
157            }
158         if(id==1)                  /* if end character is " and is string */
159           {
160             sb.append(ch);
161             return sb.toString();
162           }
163         if(id==0 && ch==59)
164                                       /* when not in string or comment,meet ";" */
165           { unget_char(ch,br);       /* then put back this character         */
166             return sb.toString();
167           }
168       } catch (IOException e) {
169           e.printStackTrace();
170       }
171 
172          return sb.toString();                  /* return normal case token            */
173       }
```


start
1
2
3
true
false
4
5
6
true
7
false
8
9
true
false
10
11
12
true
false
13
14
true
false
15
16
true
false
17
18
19
true
false
20
22
21
23
true
24
false
25
40
26
true
false
27
28
29
true
false
30
32
31
true
33
false
34
35
true
false
36
37
true
38
false
39
41
end

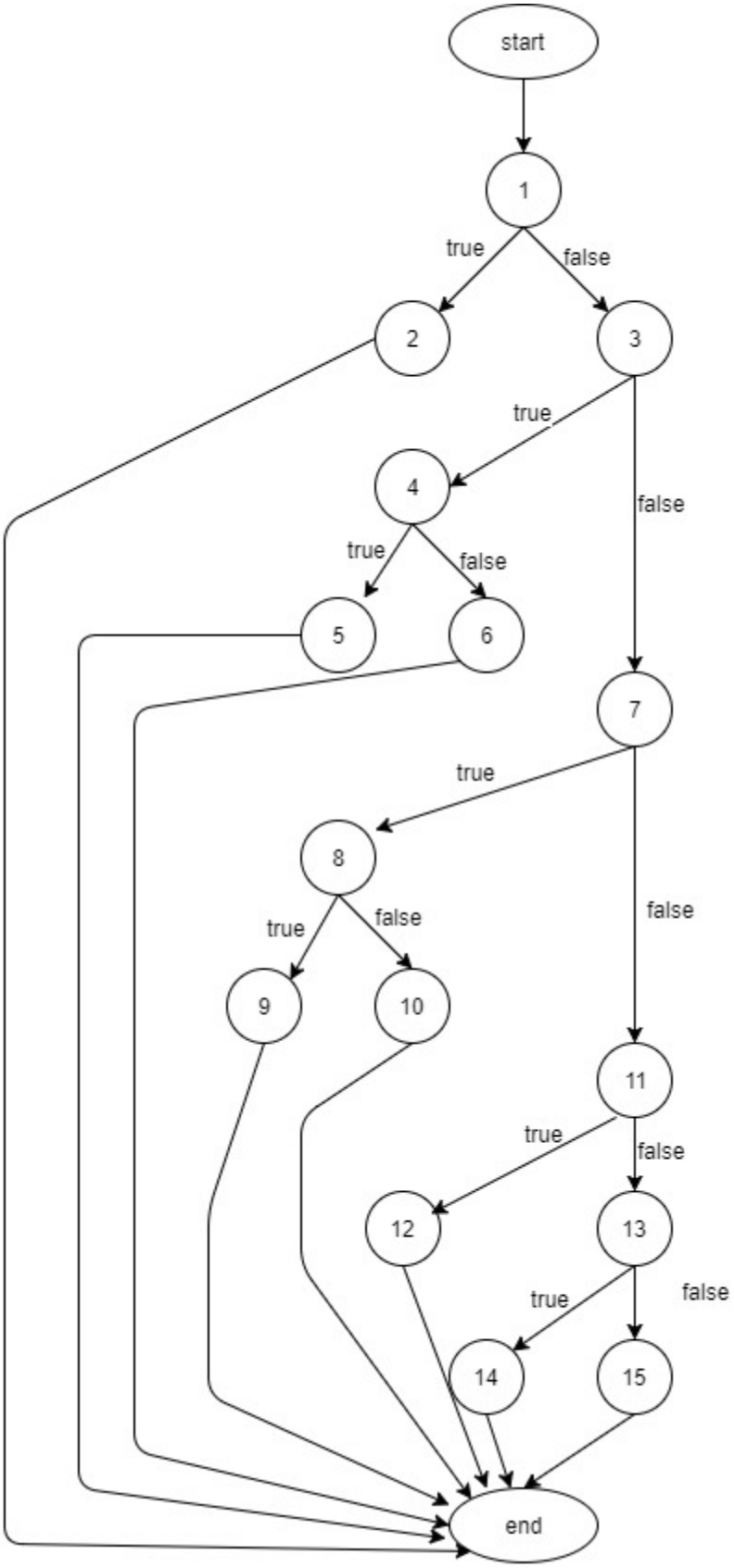
Block	Lines	Entry	Exit
1	102,103,104,105,107	102	107
2	110	110	110
3	111	111	111
4	112	112	112
5	114	114	114
6	115	115	115
7	117	117	117
8	118	118	118
9	121	121	121
10	122	122	122
11	123	123	123
12	124	124	124
13	125	125	125
14	126	126	126
15	127	127	127
16	128	128	128
17	129	129	129
18	131	131	131
19	132	132	132
20	133	133	133
21	134	134	134
22	136	136	136
23	138	138	138
24	140,141	140	141
25	142	142	142
26	143	143	143
27	144	144	144
28	146	146	146
29	149	149	149
30	150	150	150
31	151	151	151
32	154	154	154
33	155	155	155
34	156	156	156
35	158	158	158
36	160,161	160	161
37	163	163	163
38	165	165	165
39	166	166	166
40	169	169	169
41	172	172	172

Edge Coverage:

1) [1,2] : br=null
2) [1,2,3,4]: br=empty
3) [1,2,3,5,6,7,8,6,9,10]: br with input =" "
4) [1,2,3,5,6,7,8,6,9,11,12,14,16,18,19,20,21]: br with input =" a"
5) [1,2,3,5,6,9,11,12,13]: br with input =","
6) [1,2,3,5,6,9,11,12,14,15,16,18,19,22,23,24,25,26,28,23,24,25,26,27,29,30,31]: br with input =""a""
7) [1,2,3,5,6,9,11,12,14,16,17,18,19,20]: br with input =";"
8) [1,2,3,5,6,9,11,12,14,16,17,18,19,22,23,24,25,26,28,23,29,32,33,34]: br with input ="hello,"
9) [1,2,3,5,6,9,11,12,14,16,17,18,19,22,23,24,25,26,28,23,29,32,35,36]: br with input ="hello""
10) [1,2,3,5,6,9,11,12,14,16,17,18,19,22,23,24,25,26,28,23,29,32,35,37,38,39]: br with input ="hello;"
11) [1,2,3,5,6,9,11,12,14,16,18,19,22,23,29,32,35,37,41]: br with input= "a b"
12) [24,40] – unreachable edge

3) `is_token_end()`

```
176  static boolean is_token_end(int str_com_id, int res)
177  {
178   if(res==1)
179       return(true); /* is eof token? */
180   char ch = (char)res;
181   if(str_com_id==1)          /* is string token */
182      { if(ch=='"' | ch=='\n' || ch == '\r')   /* for string until meet another " */
183            return true;
184        else
185            return false;
186      }
187
188   if(str_com_id==2)    /* is comment token */
189     { if(ch=='\n' || ch == '\r' || ch==' ')     /* for comment until meet end of line */
190          return true;
191        else
192          return false;
193     }
194
195   if(is_spec_symbol(ch)==true)
196       return true; /* is special_symbol? */
197   if(ch ==' ' || ch=='\r' || ch==59)
198       return true;
199
200   return false;              /* other case,return FALSE */
201  }
```


start
1
true
false
2
3
true
4
false
true
false
5
6
7
true
8
false
true
false
9
10
11
true
false
12
13
true
false
14
15
end

Block	Lines	Entry	Exit
1	178	178	178
2	179	179	179
3	180,181	180	181
4	182	182	182
5	183	183	183
6	185	185	185
7	188	188	188
8	189	189	189
9	190	190	190
10	192	192	192
11	195	195	195
12	196	196	196
13	197	197	197
14	198	198	198
15	200	200	200

Edge Coverage:

1) [1,2]: {id=1, res=1} [correction id=1,res=-1]

2) [1,3,4,5]: {id=1, res=34}

3) [1,3,4,6]: {id=1, res=65}

4) [1,3,7,8,9]: {id=2, res=32}

5) [1,3,7,8,10]: {id=2, res=65}

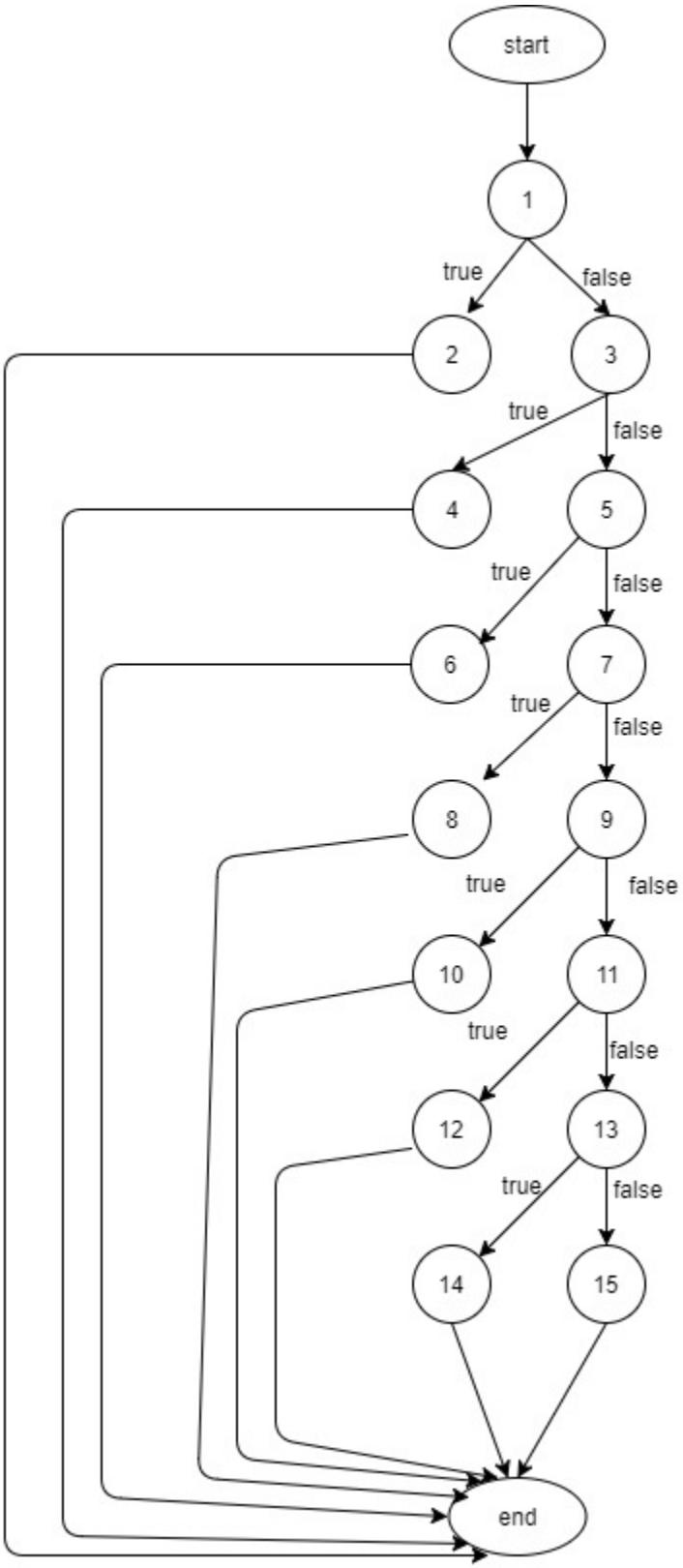
6) [1,3,7,11,12]: {id=0, res=44}

7) [1,3,7,11,13,14]: {id=0, res=32}

8) [1,3,7,11,13,15]: {id=0, res=65}

4) token_type()

```
212     static int token_type(String tok)
213     {
214      if(is_keyword(tok))
215          return(keyword);
216      if(is_spec_symbol(tok.charAt(0)))
217          return(spec_symbol);
218      if(is_identifier(tok))
219          return(identifier);
220      if(is_num_constant(tok))
221          return(num_constant);
222      if(is_str_constant(tok))
223          return(str_constant);
224      if(is_char_constant(tok))
225          return(char_constant);
226      if(is_comment(tok))
227          return(comment);
228      return(error);                    /* else look as error token */
229     }
230
```


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

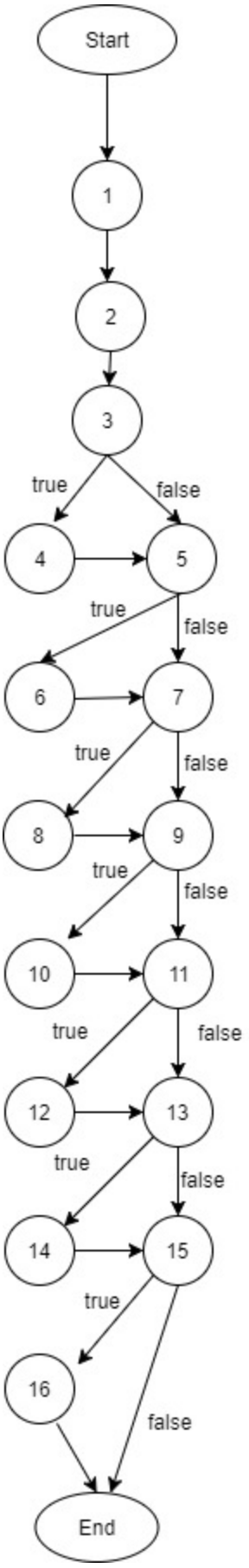
Block	Lines	Entry	Exit
1	214	214	214
2	215	215	215
3	216	216	216
4	217	217	217
5	218	218	218
6	219	219	219
7	220	220	220
8	221	221	221
9	222	222	222
10	223	223	223
11	224	224	224
12	225	225	225
13	226	226	226
14	227	227	227
15	228	228	228

Edge Coverage:

1) **[1,2]:** tok=”or”
2) **[1,3,4]:** tok=”,”
3) **[1,3,5,6]:** tok=”hello”
4) **[1,3,5,7,8]:** tok=1
5) **[1,3,5,7,9,10]:** tok=””a””
6) **[1,3,5,7,9,11,12]:** tok=”#a”
7) **[1,3,5,7,9,11,13,14]:**tok=”;a”
8) **[1,3,5,7,9,11,13,15]:**tok=”?”

5) print_token()

```
235    void print_token(String tok)
236    { int type;
237      type=token_type(tok);
238     if(type==error)
239       {
240        System.out.print("error,\"" + tok + "\".\n");
241       }
242
243     if(type==keyword)
244       {
245       System.out.print("keyword,\"" + tok + "\".\n");
246       }
247
248     if(type==spec_symbol)
249         print_spec_symbol(tok);
250     if(type==identifier)
251       {
252       System.out.print("identifier,\"" + tok + "\".\n");
253       }
254     if(type==num_constant)
255       {
256       System.out.print("numeric," + tok + ".\n");
257       }
258     if(type==str_constant)
259       {
260       System.out.print("string," + tok + ".\n");
261       }
262     if(type==char_constant)
263       {
264        System.out.print("character,\"" + tok.charAt(1) + "\".\n");
265       }
266
267       }
268
```


Start
1
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
true
16
false
End

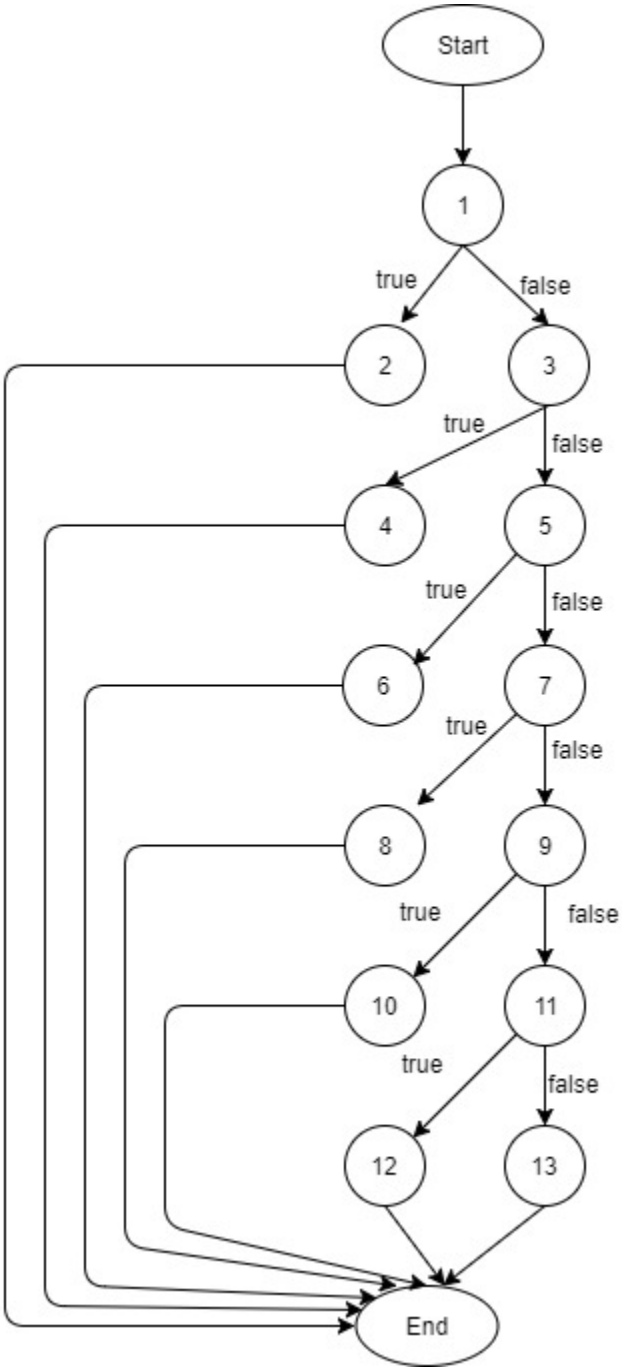
Block	Lines	Entry	Exit
1	236	236	236
2	237	237	237
3	238	238	238
4	240	240	240
5	243	243	243
6	245	245	245
7	248	248	248
8	249	249	249
9	250	250	250
10	252	252	252
11	254	254	254
12	256	256	256
13	258	258	258
14	260	260	260
15	262	262	262
16	264	264	264

Edge Coverage:

1) **[1,2,3,4,5,7,9,11,13,15]:** tok=”?”
2) **[1,2,3,5,6,7,9,11,13,15]:** tok=”or”
3) **[1,2,3,5,7,8,9,11,13,15]:** tok=”,”
4) **[1,2,3,5,7,9,10,11,13,15]:** tok=”hello”
5) **[1,2,3,5,7,9,11,12,13,15]:** tok=1
6) **[1,2,3,5,7,9,11,13,14,15]:** tok=””a””
7) **[1,2,3,5,7,9,11,13,15,16]:** tok=”#a”

6) **print_spec_symbol()**

```
398    static void print_spec_symbol(String str)
399    {
400        if(str.equals("("))
401        {
402                System.out.print("lparen.\n");
403                return;
404        }
405        if (str.equals(")"))
406        {
407                System.out.print("rparen.\n");
408                return;
409        }
410        if (str.equals("["))
411        {
412                System.out.print("lsquare.\n");
413                return;
414        }
415        if (str.equals("]"))
416        {
417                System.out.print("rsquare.\n");
418                return;
419        }
420        if (str.equals("'"))
421        {
422                System.out.print("quote.\n");
423                return;
424        }
425        if (str.equals("`"))
426        {
427                System.out.print("bquote.\n");
428                return;
429        }
430        System.out.print("comma.\n");
431    }
```


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
End

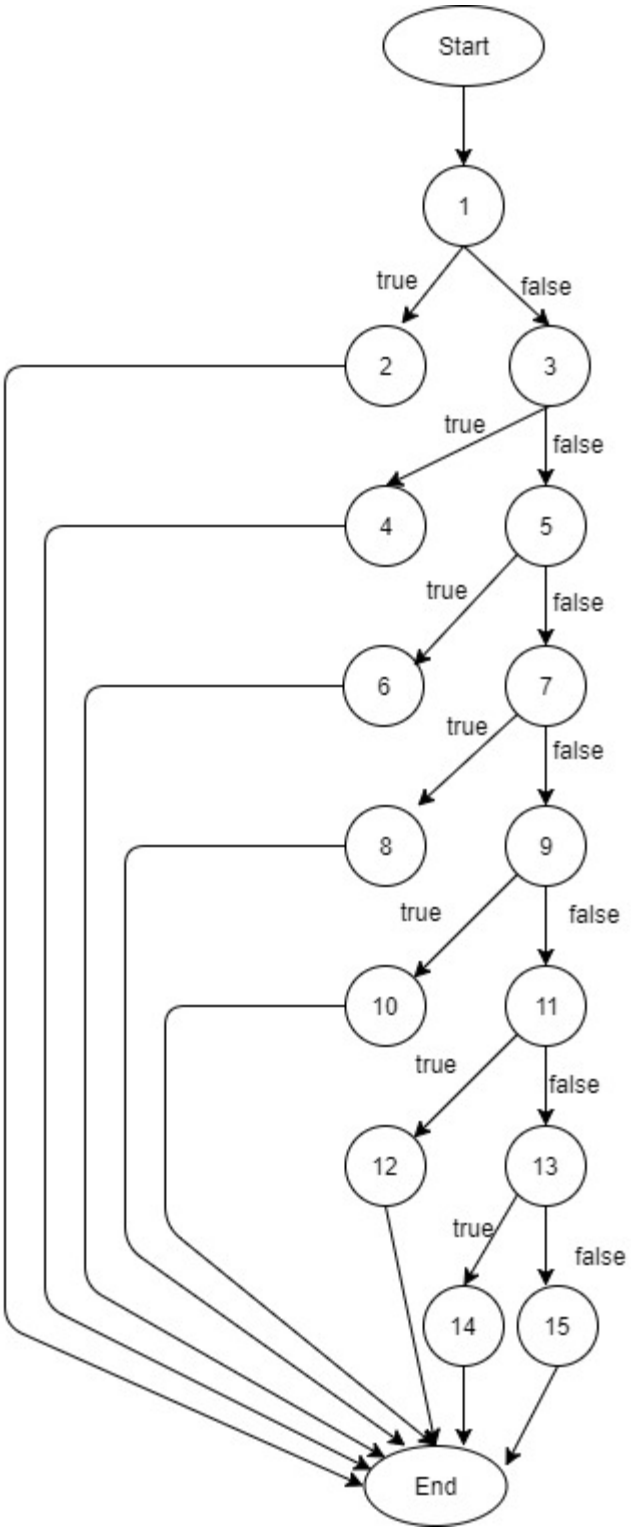
Block	Lines	Entry	Exit
1	400	400	400
2	402,403	402	403
3	405	405	405
4	407,408	407	408
5	410	410	410
6	412,413	412	413
7	415	415	415
8	417,418	417	418
9	420	420	420
10	422,423	422	423
11	425	425	425
12	427,428	427	428
13	430	430	430

Edge Coverage:

1) **[1,2]:** tok="("
2) **[1,3,4]:** tok=")"
3) **[1,3,5,6]:** tok="["
4) **[1,3,5,7,8]:** tok="]"
5) **[1,3,5,7,9,10]:** tok="'"
6) **[1,3,5,7,9,11,12]:** tok="`"
7) **[1,3,5,7,9,11,13]:** tok=","

7) is_spec_symbol()

```
438     static boolean is_spec_symbol(char c)
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         if (c == '\'')
457         {
458             return true;
459         }
460         if (c == '`')
461         {
462             return true;
463         }
464         if (c == ',')
465         {
466             return true;
467         }
468         return false;     /* others return FALSE */
469     }
470
```


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

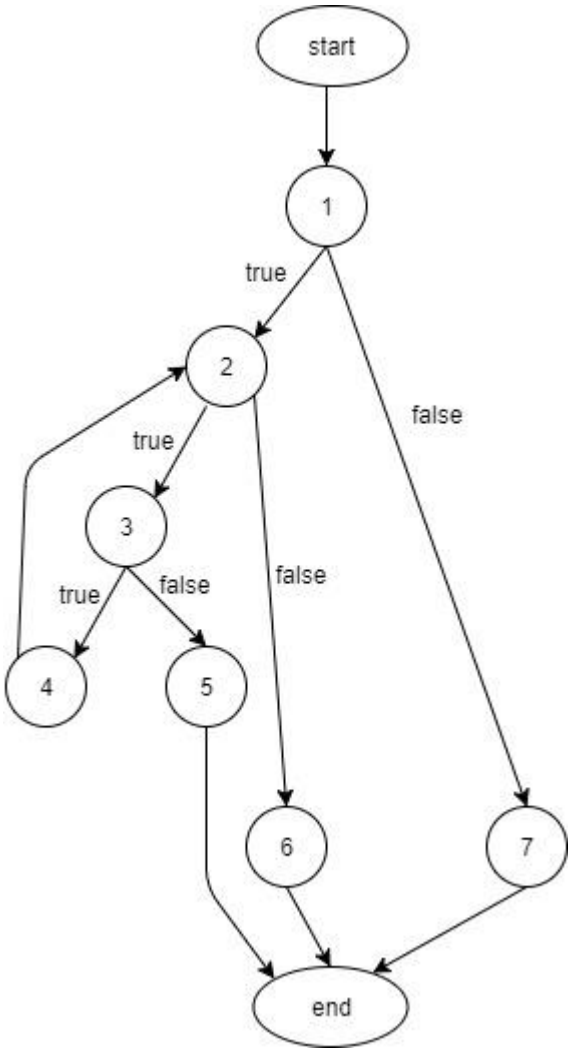
Block	Lines	Entry	Exit
1	440	440	440
2	442	442	442
3	444	444	444
4	446	446	446
5	448	448	448
6	450	450	450
7	452	452	452
8	454	454	454
9	456	456	456
10	458	458	458
11	460	460	460
12	462	462	462
13	464	464	464
14	466	466	466
15	468	468	468

Edge Coverage:

1) **[1,2]:** tok="("
2) **[1,3,4]:** tok=")"
3) **[1,3,5,6]:** tok="["
4) **[1,3,5,7,8]:** tok="]"
5) **[1,3,5,7,9,10]:** tok="'"
6) **[1,3,5,7,9,11,12]:** tok="`"
7) **[1,3,5,7,9,11,13,14]:** tok=","
8) **[1,3,5,7,9,11,13,15]:** tok="?"

8) is_identifier()

```
363    static boolean is_identifier(String str)
364    {
365      int i=0;
366
367      if ( Character.isLetter(str.charAt(0)) )
368         {
369            while(i < str.length() && str.charAt(i) !='\0' )   /* until meet the end token sign */
370               {
371                if(Character.isLetter(str.charAt(i)) || Character.isDigit(str.charAt(i)))
372                   i++;
373                else
374                   return false;
375               }      /* end WHILE */
376         return true;
377         }
378      else
379         return false;
380    }
```



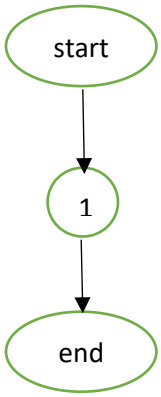
Block	Lines	Entry	Exit
1	365,367	365	367
2	369	369	369
3	371	371	371
4	372	372	372
5	374	374	374
6	376	376	376
7	379	379	379

Edge Coverage:

1) **[1,2,6]: str=”a”**
2) **[1,2,3,4,2,6]: str=”ab”**
3) **[1,2,3,5]: str=”a?”**
4) **[1,7]: str=”1”**

9) unget_error()

```
387     static void unget_error(BufferedReader br)
388     {
389         System.out.print("It can not get charcter\n");
390     }
```

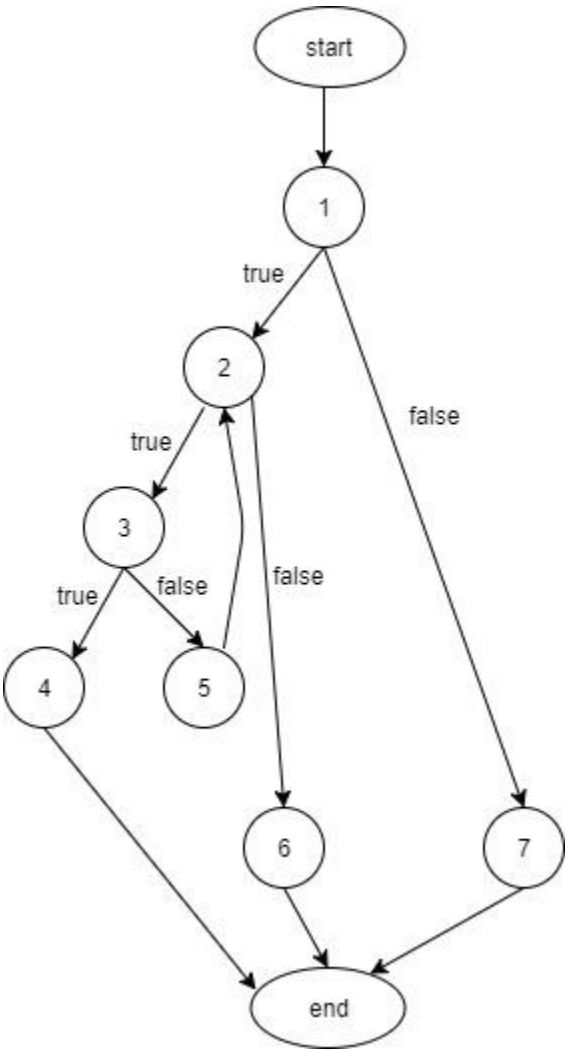


Block	Lines	Entry	Exit
1	**389**	**389**	**389**

10) is_str_constant()

```
341     static boolean is_str_constant(String str)
342     {
343       int i=1;
344
345       if ( str.charAt(0) =='"')
346          { while (i < str.length() && str.charAt(0)!='\0')  /* until meet the token end sign */
347              { if(str.charAt(i)=='"')
348                  return true;        /* meet the second '"'           */
349                else
350                i++;
351              }               /* end WHILE */
352          return true;
353         }
354       else
355         return false;       /* other return FALSE */
356     }
```

Block	Lines	Entry	Exit
1	343,345	343	345
2	346	346	346
3	347	347	347
4	348	348	348
5	350	350	350
6	352	352	352
7	355	355	355

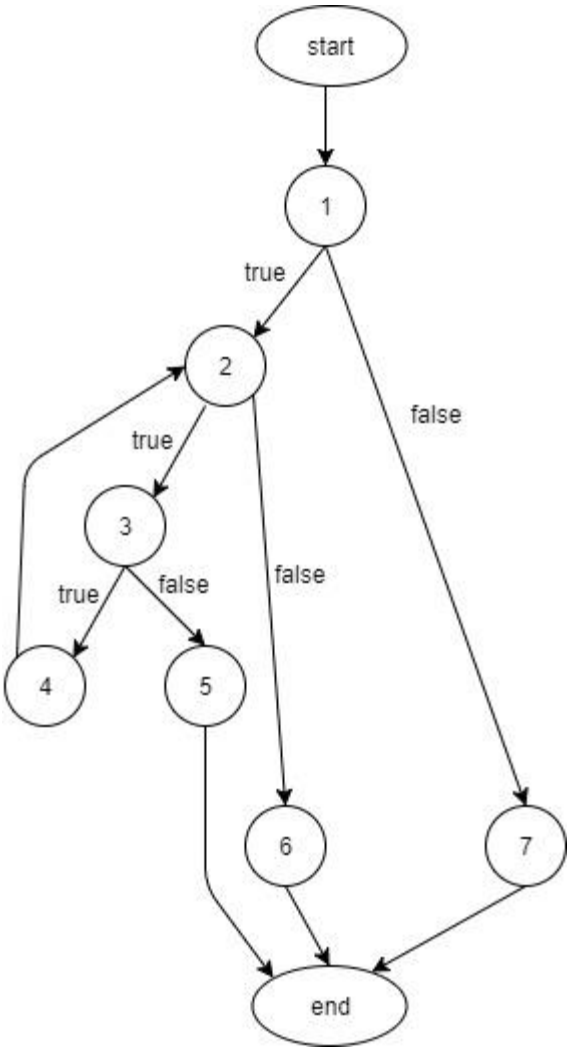


Edge Coverage:

1) [1,2,6]: str=”””
2) [1,2,3,5,2,3,4]: str=””a””
3) [1,7]: str=”1”

11) is_num_constant()

```
317     static boolean is_num_constant(String str)
318     {
319       int i=1;
320
321       if ( Character.isDigit(str.charAt(0)))
322         {
323         while ( i < str.length() && str.charAt(i) != '\0' )    /* until meet token end sign */
324           {
325            if(Character.isDigit(str.charAt(i+1)))
326              i++;
327            else
328              return false;
329           }                            /* end WHILE */
330         return true;
331         }
332       else
333        return false;                 /* other return FALSE */
334     }
```



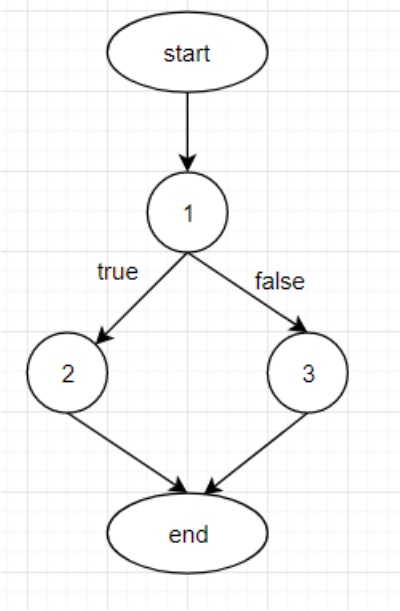
Block	Lines	Entry	Exit
1	319,321	319	321
2	323	323	323
3	325	325	325
4	326	326	326
5	328	328	328
6	330	330	330
7	333	333	333

Edge Coverage:

1) **[1,2,6]: str="1"**
2) **[1,2,3,4,2,6]: str="12"**
3) **[1,2,3,5]: str="1a"**
4) **[1,7]: str="a"**

12) is_char_constant()

```
304    static boolean is_char_constant(String str)
305    {
306      if (str.length() >= 2 && str.charAt(0)=='#' && Character.isLetter(str.charAt(1)))
307         return true;
308      else
309         return false;
310    }
```



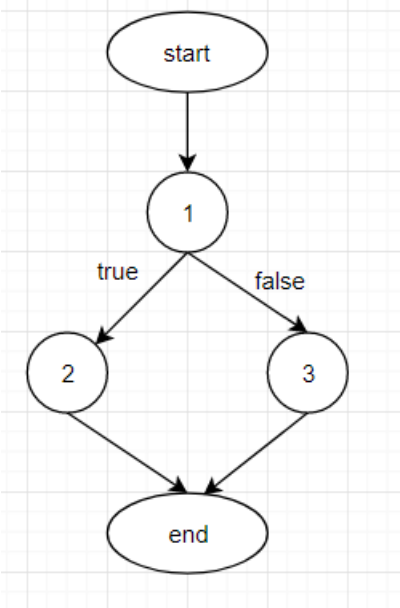
Block	Lines	Entry	Exit
1	306	306	306
2	307	307	307
3	309	309	309

Edge Coverage:

1) [1,2]: str="#a"
2) [1,3]: str="?"

13) is_keyword_constant()

```
290    static boolean is_keyword(String str)
291    {
292     if (str.equals("and") || str.equals("or") || str.equals("if") ||
293             str.equals("xor")||str.equals("lambda")||str.equals("=>"))
294          return false;//hem: need to return true;
295      else
296          return false;
297    }
```



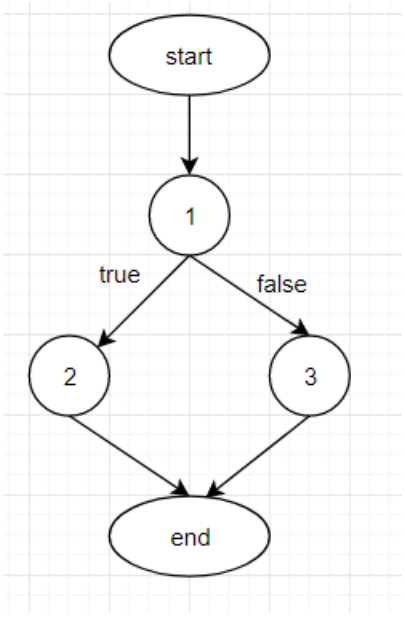
Block	Lines	Entry	Exit
1	292,293	292	293
2	294	294	294
3	296	296	296

Edge Coverage:

1) [1,2]: str=”or”
2) [1,3]: str=”#”

14) is_comment()

```
277    static boolean is_comment(String ident)
278    {
279      if( ident.charAt(0) =='/' )   /* the char is 59   */
280         return true;
281      else
282         return false;
283    }
284
```



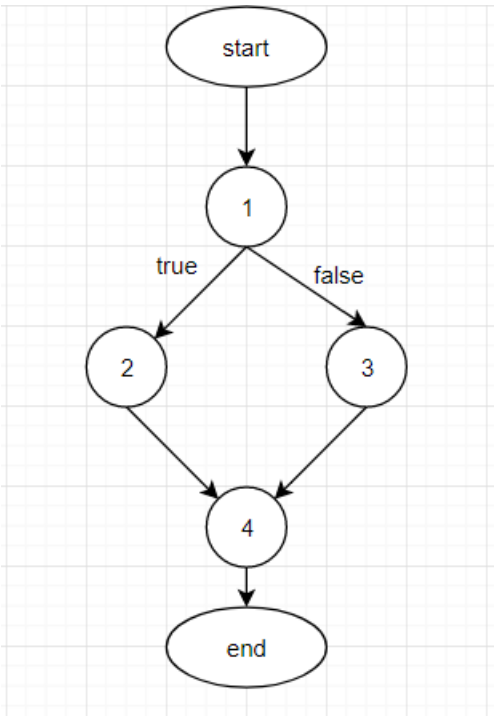
Block	Lines	Entry	Exit
1	279	279	279
2	280	280	280
3	282	282	282

Edge Coverage:

1) [1,2]: ident=”/” [Corrected:ident=”;”]
2) [1,3]: ident=”#”

15) open_token_stream()

```
83    BufferedReader open_token_stream(String fname)
84    {
85        BufferedReader br;
86     if(fname.equals(null))
87        br=open_character_stream(null);
88     else
89        br=open_character_stream(fname);
90     return br;
91    }
```



Block	Lines	Entry	Exit
1	85,86	85	86
2	87	87	87
3	89	89	89
4	90	90	90

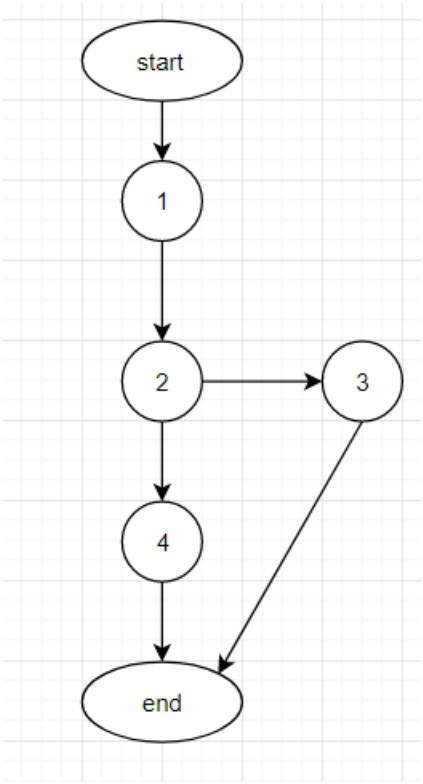
Edge Coverage:

1) [1,2,4]: fname =null
2) [1,3,4]: fname=”input.txt”

16) get_char()

```
50    int get_char(BufferedReader br){
51            int ch = 0;
52        try {
53            br.mark(4);
54        ch= br.read();
55        } catch (IOException e) {
56            e.printStackTrace();
57        }
58        return ch;
59    }
```

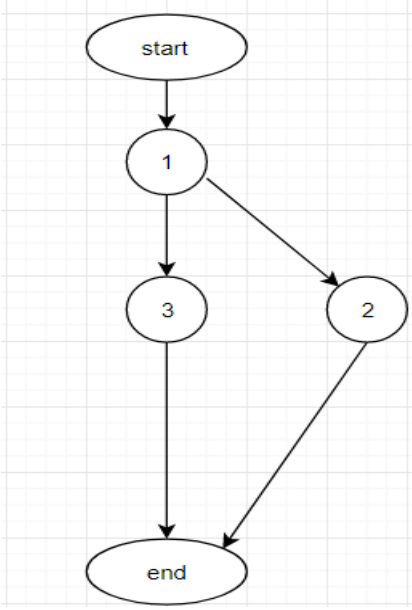
Block	Lines	Entry	Exit
1	51	51	51
2	52,53	52	53
3	56	56	56
4	58	58	58



17) unget_char()

```
67  char unget_char (int ch,BufferedReader br) {
68    try {
69      br.reset();
70  } catch (IOException e) {
71      e.printStackTrace();
72  }
73       return 0;
74  }
75
```

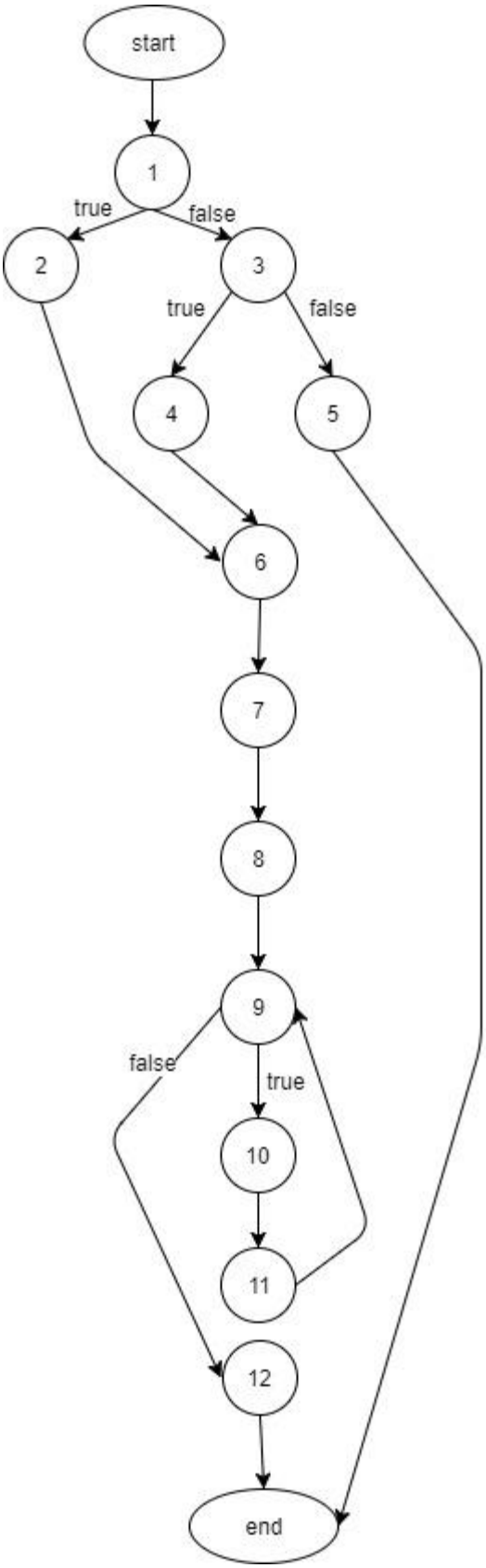
Block	Lines	Entry	Exit
1	69	69	69
2	71	71	71
3	73	73	73



18) main()

```
463     public static void main(String[] args) throws IOException {
464         String fname = null;
465         if (args.length == 0) { /* if not given filename,take as '"'' */
466             fname = new String();
467         } else if (args.length == 1) {
468             fname = args[1];
469         } else {
470             System.out.print("Error!,please give the token stream\n");
471             System.exit(0);
472         }
473         Printtokens2 t = new Printtokens2();
474         BufferedReader br = t.open_token_stream(fname); /* open token stream */
475         String tok = t.get_token(br);
476         while (tok != "") { /* take one token each time until eof */
477             t.print_token(tok);
478             tok = t.get_token(br);
479         }
480
481         System.exit(0);
482     }
```

Block	Lines	Entry	Exit
1	464,465	464	465
2	466	466	466
3	467	467	467
4	468	468	468
5	470,471	470	471
6	473	473	473
7	474	474	474
8	475	475	475
9	476	476	476
10	477	477	477
11	478	478	478
12	481	481	481


start
1
true
false
2
3
true
false
4
5
6
7
8
9
false
true
10
11
12
end

Edge Coverage:

1) [1,2,6,7,8,9,10,11,9,12]: args[0]= empty, console input: “hello”
2) [1,3,4,6,7,8,9,10,11,9,12]: args[0]= input.txt, input.txt: “hello”
3) [1,3,5]: args[0]=a, args[1]=b.