

CSE 4321 Project

Control Flow Graph

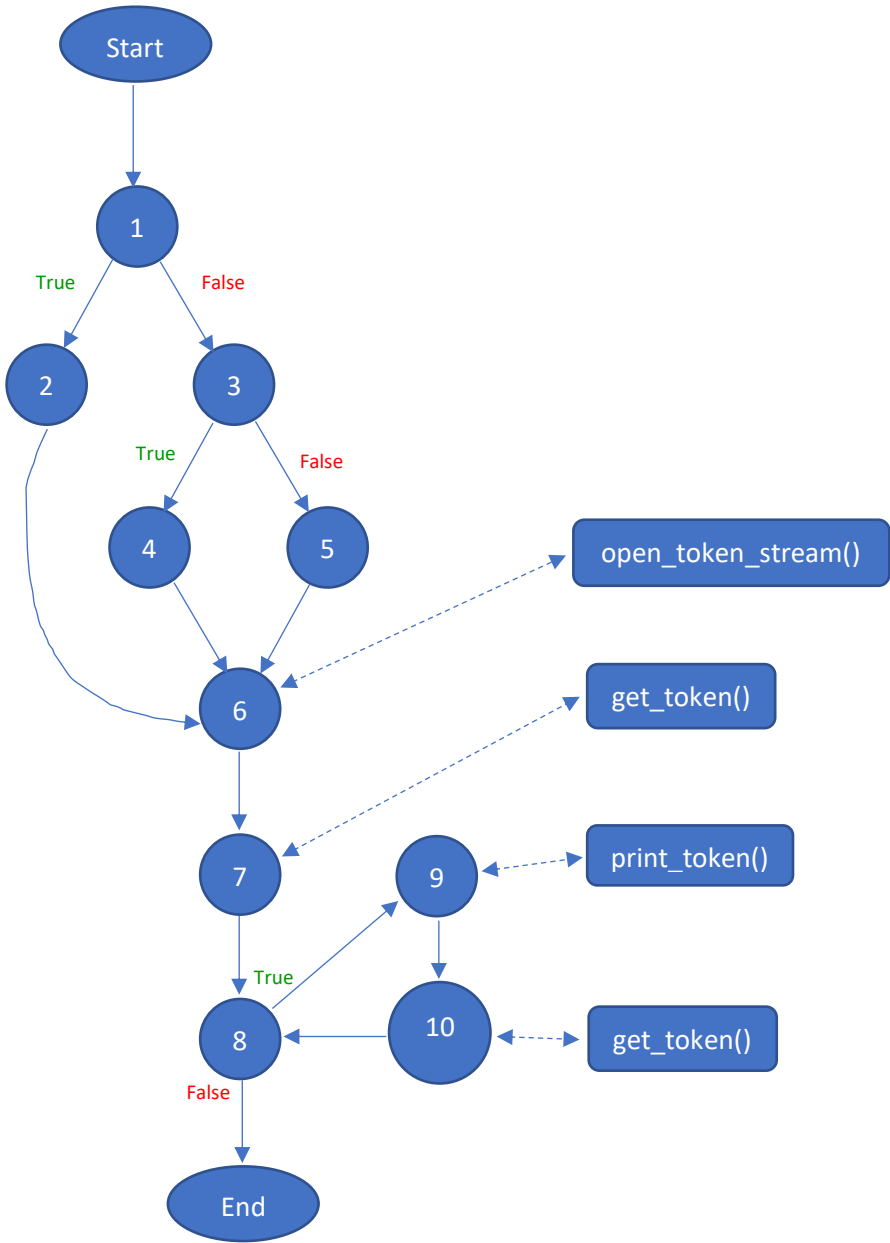
1000867697

Andrew Duong

main()

```
1 public static void main(String[] args) throws IOException {
2     String fname = null;
3     if (args.length == 0) {          /* if not given filename,take as '' */
4         fname = new String();
5     } else if (args.length == 1) {
6         fname = args[1];
7     } else {
8         System.out.print("Error!,please give the token stream\n");
9         System.exit(0);
10    }
11    Printtokens t = new Printtokens();
12    BufferedReader br = t.open_token_stream(fname); /* open token stream */
13    String tok = t.get_token(br);
14    while (tok != null) { /* take one token each time until eof */
15        t.print_token(tok);
16        tok = t.get_token(br);
17    }
18    System.exit(0);
19 }
```

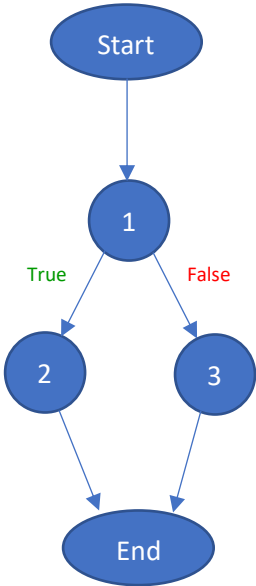
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	5	5	5
4	6	6	6
5	8, 9	8	9
6	10, 11	10	11
7	12	12	12
8	13	13	13
9	14	14	14
10	15	15	15



open_character_stream()

```
1 BufferedReader open_character_stream(String fname) {
2   BufferedReader br = null;
3   if (fname == null) {
4     br = new BufferedReader(new InputStreamReader(System.in));
5   } else {
6     try {
7       FileReader fr = new FileReader(fname);
8       br = new BufferedReader(fr);
9     } catch (FileNotFoundException e) {
10      System.out.print("The file " + fname + " doesn't exists\n");
11      e.printStackTrace();
12    }
13  }
14  return null;
15 }
```

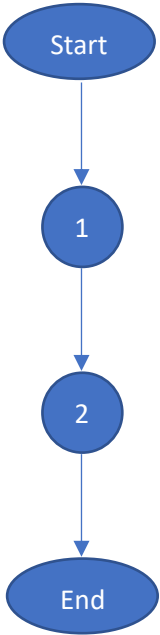
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	6, 7, 8, 9, 10, 11	6	11



get_char()

```
1 int get_char(BufferedReader br){
2     int ch = 0;
3     try {
4         br.mark(3);
5         ch= br.read();
6     } catch (IOException e) {
7         e.printStackTrace();
8     }
9     return ch;
10 }
```

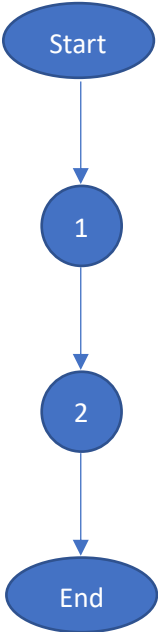
Block	Lines	Entry	Exit
1	2, 3, 4, 5, 6, 7	2	7
2	8	8	8



unget_char()

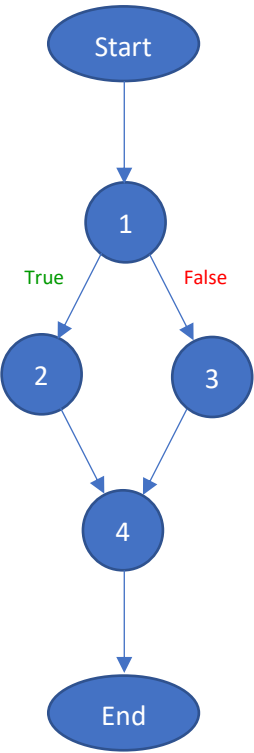
```
1 char unget_char (int ch,BufferedReader br) {
2     try {
3         br.reset();
4     } catch (IOException e) {
5         e.printStackTrace();
6     }
7     return 0;
8 }
```

Block	Lines	Entry	Exit
1	2, 3, 4, 5	2	5
2	6	6	6



open_token_stream()

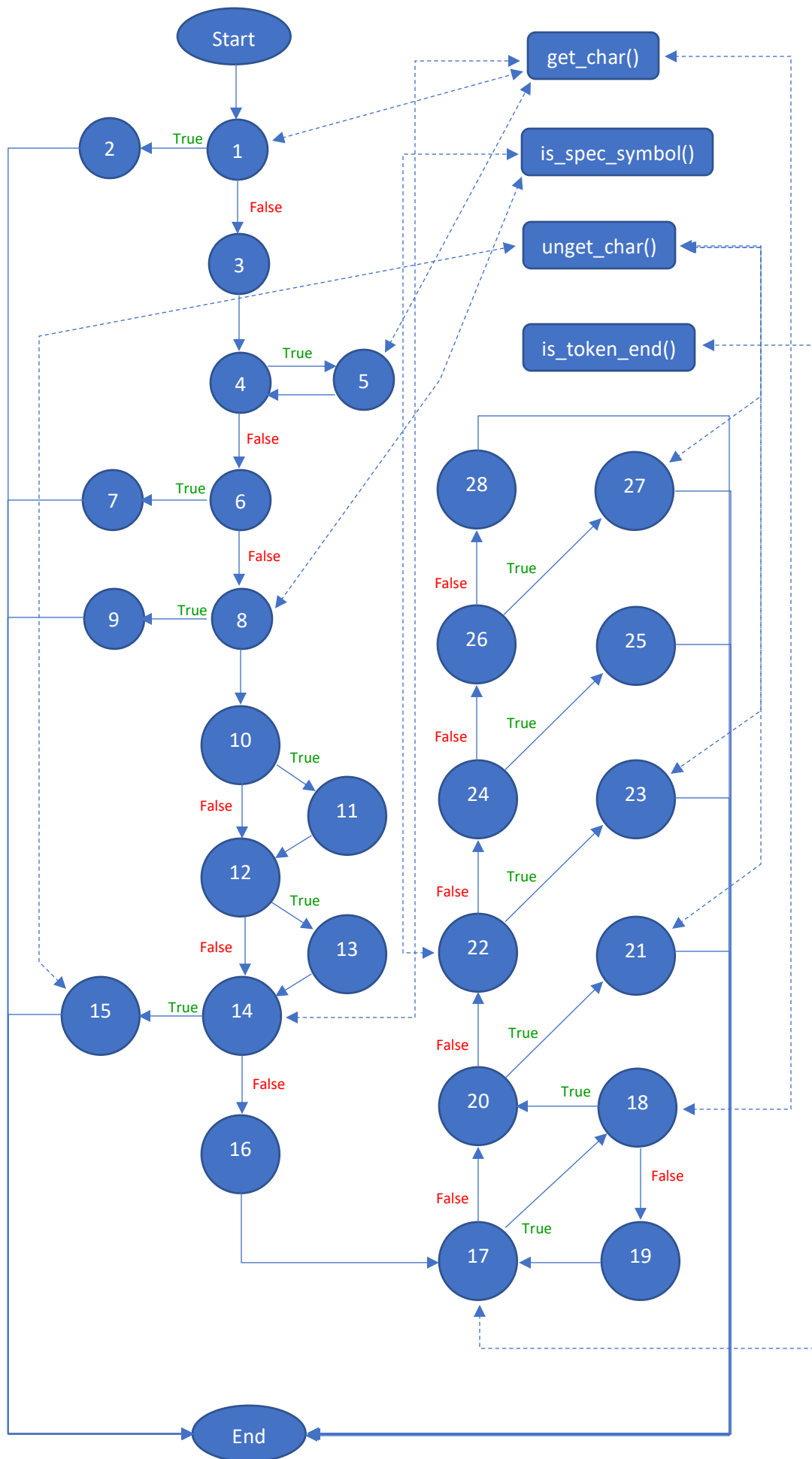
<pre>1 BufferedReader open_token_stream(String fname) { 2 BufferedReader br; 3 if(fname.equals(null)) 4 br=open_character_stream(null); 5 else 6 br=open_character_stream(fname); 7 return br; }</pre>			
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	6	6	6
4	7	7	7



get_token()

```
1 String get_token(BufferedReader br)
2 {
3     int i=0,j;
4     int id=0;
5     int res = 0;
6     char ch = '\0';
7
8     StringBuilder sb = new StringBuilder();
9
10    try {
11        res = get_char(br);
12        if (res == -1) {
13            return null;
14        }
15        ch = (char)res;
16        while(ch=='\t' || ch=='\n' || ch == '\r') /* strip all blanks
17            until meet characters */
18        {
19            res = get_char(br);
20            ch = (char)res;
21        }
22
23        if(res == -1) return null;
24        sb.append(ch);
25        if(is_spec_symbol(ch)==true) return sb.toString();
26        if(ch=='"') id=2; /* prepare for string */
27        if(ch=='#') id=1; /* prepare for comment */
28
29        res = get_char(br);
30        if (res == -1) {
31            unget_char(ch,br);
32            return sb.toString();
33        }
34        ch = (char)res;
35
36        while (is_token_end(id,res) == false)/* until meet the end
37            character */
38        {
39            sb.append(ch);
40            br.mark(4);
41            res = get_char(br);
42            if (res == -1) {
43                break;
44            }
45            ch = (char)res;
46
47            if(res == -1) /* if end character is eof token */
48            { unget_char(ch,br); /* then put back eof on token_stream */
49              return sb.toString();
50            }
51
52            if(is_spec_symbol(ch)==true) /* if end character is
53                special_symbol */
54            { unget_char(ch,br); /* then put back this character
55                */
56              return sb.toString();
57            }
58
59            if(id==1) /* if end character is " and is string
60                */
61            {
62                sb.append(ch);
63                return sb.toString();
64            }
65
66            if(id==0 && ch=='#') /* when not in string or
67                comment,meet ";" */
68            { unget_char(ch,br); /* then put back this character
69                */
70              return sb.toString();
71            }
72
73        } catch (IOException e) {
74            e.printStackTrace();
75        }
76
77    }
78    return sb.toString(); /* return normal case token
79    */
80 }
```

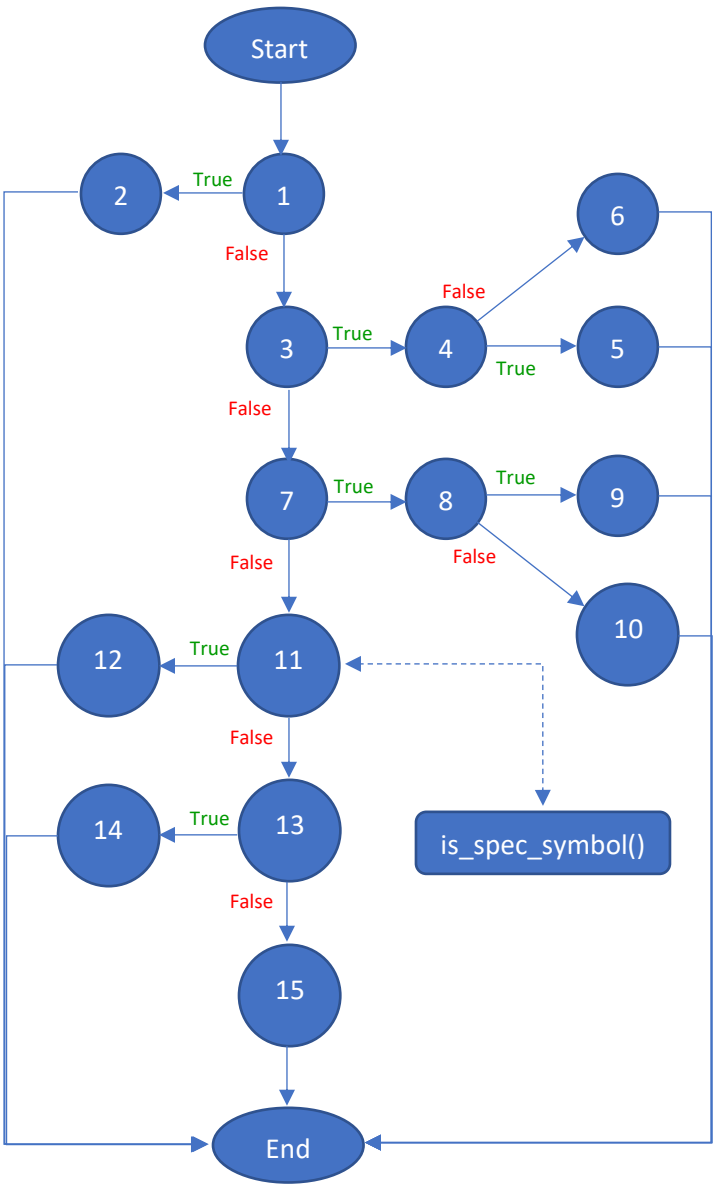
Block	Lines	Entry	Exit
1	2, 3, 4, 5, 6, 7, 8, 9	2	9
2	10	10	10
3	11	11	11
4	12	12	12
5	13, 14	13	14
6	15	15	15
7	16	16	16
8	17, 18	17	18
9	19	19	19
10	20	20	20
11	21	21	21
12	22	22	22
13	23	23	23
14	24, 25	24	25
15	26, 27	26	27
16	28	28	28
17	29	29	29
18	30, 31, 32, 33	30	33
19	35	35	35
20	36	36	36
21	37, 38	37	38
22	39	39	39
23	40, 41	40	41
24	42	42	42
25	43, 44	43	44
26	45	45	45
27	46, 47	46	47
28	48, 49, 50	48	50



is_token_end()

```
1 static boolean is_token_end(int str_com_id, int res)
2 {
3     if(res== -1) return true; /* is eof token? */
4     char ch = (char)res;
5     if(str_com_id==1) /* is string token */
6     { if(ch=='"' || ch=='\n' || ch=='\r') /* for string until meet
7       another " */
8         return true;
9     }
10    else
11        return false;
12
13    if(str_com_id==2) /* is comment token */
14    { if(ch=='\n' || ch=='\r' || ch=='\t') /* for comment until meet
15      end of line */
16        return true;
17    }
18    else
19        return false;
20
21    if(is_spec_symbol(ch)==true) return true; /* is special_symbol? */
22    if(ch==' ' || ch=='\n' || ch=='\r' || ch==59) return true;
23
24    return false; /* other case,return FALSE */
25 }
```

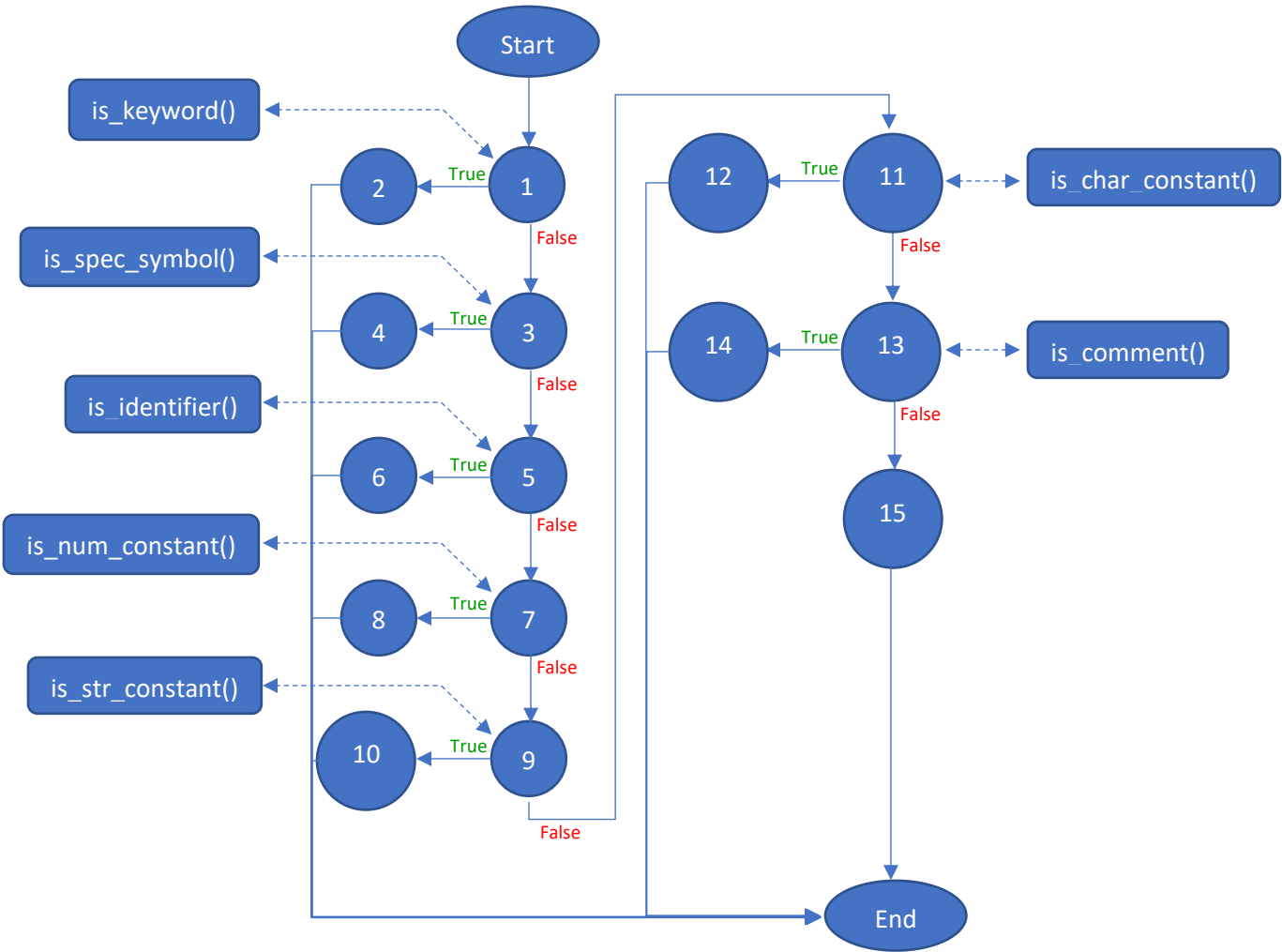
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4, 5	4	5
4	6	6	6
5	7	7	7
6	9	9	9
7	10	10	10
8	11	11	11
9	12	12	12
10	14	14	14
11	15	15	15
12	16	16	16
13	17	17	17
14	18	18	18
15	19	19	19



token_type()

```
1 static int token_type(String tok)
2 {
3   if(is_keyword(tok)) 3 return(keyword);
4   if(is_spec_symbol(tok.charAt(0))) 5 return(spec_symbol);
5   if(is_identifier(tok)) 7 return(identifier);
6   if(is_num_constant(tok)) 9 return(num_constant);
7   if(is_str_constant(tok)) 11 return(str_constant);
8   if(is_char_constant(tok)) 13 return(char_constant);
9   if(is_comment(tok)) 15 return(comment);
10  return(error); /* else look as error token */
11 }
```

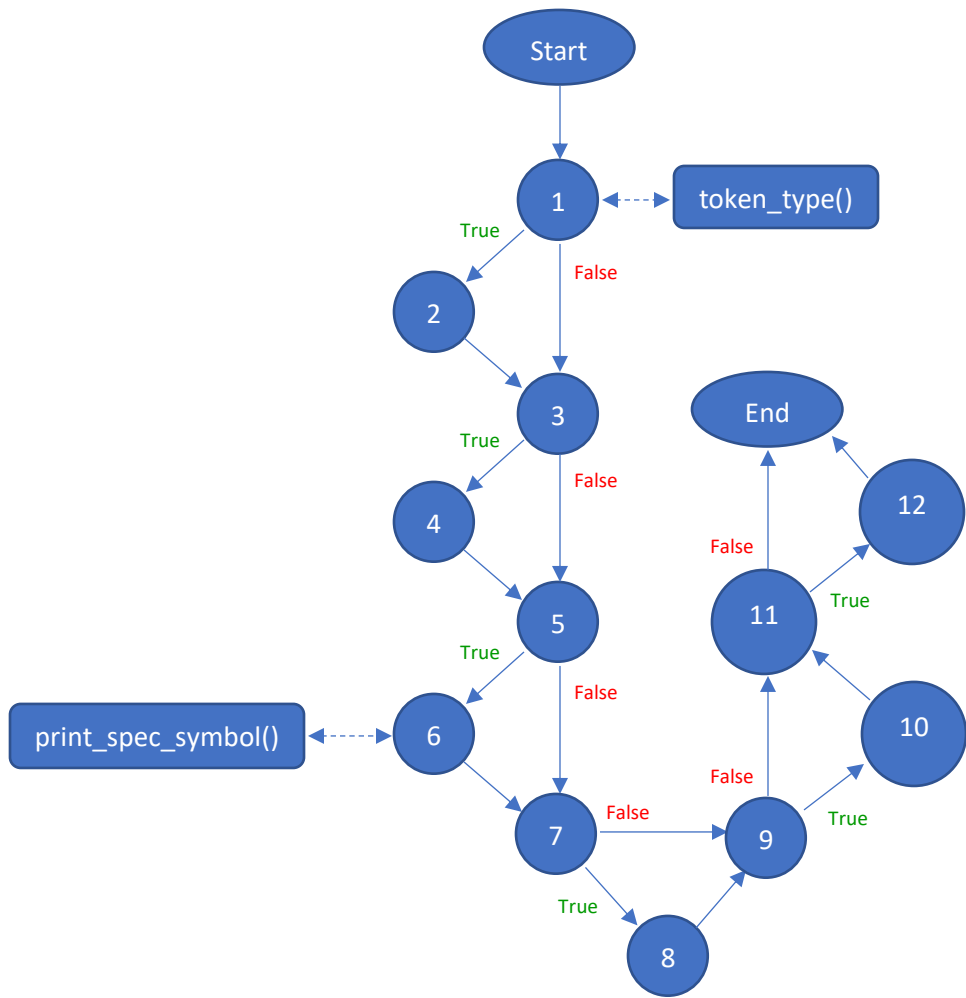
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	7	7	7
7	8	8	8
8	9	9	9
9	10	10	10
10	11	11	11
11	12	12	12
12	13	13	13
13	14	14	14
14	15	15	15
15	16	16	16



print_token()

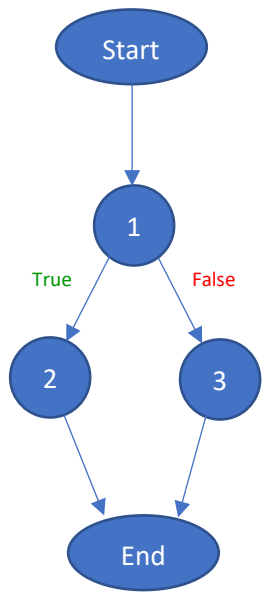
```
1 void print_token(String tok)
2 { int type;
3   type=token_type(tok);
4   if(type==error)
5   {
6     System.out.print("error,\"" + tok + "\".\n");
7   }
8   if(type==keyword)
9   {
10    System.out.print("keyword,\"" + tok + "\".\n");
11  }
12  if(type==spec_symbol) print_spec_symbol(tok);
13  if(type==identifier)
14  {
15    System.out.print("identifier,\"" + tok + "\".\n");
16  }
17  if(type==num_constant)
18  {
19    System.out.print("numeric," + tok + ".\n");
20  }
21  if(type==char_constant)
22  {
23    System.out.print("character,\"" + tok.charAt(1) + "\".\n");
24  }
25 }
```

Block	Lines	Entry	Exit
1	2, 3, 4	2	4
2	6	6	6
3	7	7	7
4	8	8	8
5	9	9	9
6	10	10	10
7	11	11	11
8	12	12	12
9	13	13	13
10	14	14	14
11	15	15	15
12	16	16	16



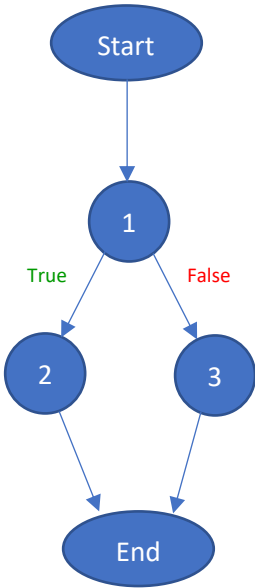
is_comment()

<pre>1 static boolean is_comment(String ident) { 2 if(ident.charAt(0) ==59) /* the char is 59 */ 3 return true; 4 else 5 return false; }</pre>			
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	5	5	5



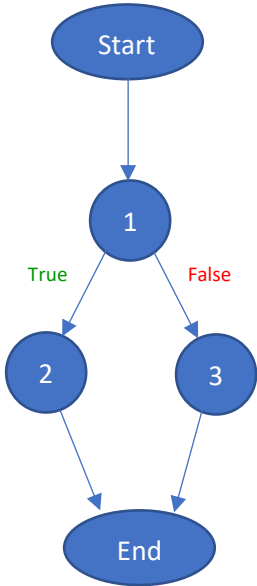
is_keyword()

<pre>1 static boolean is_keyword(String str) 2 { 3 if (str.equals("and") str.equals("or") str.equals("if") 4 str.equals("xor") str.equals("lambda") str.equals("=>")) 5 return true; 6 else 7 return false; 8 }</pre>					
Block	Lines	Entry	Exit		
1	2	2	2		
2	3	3	3		
3	5	5	5		



is_char_constant()

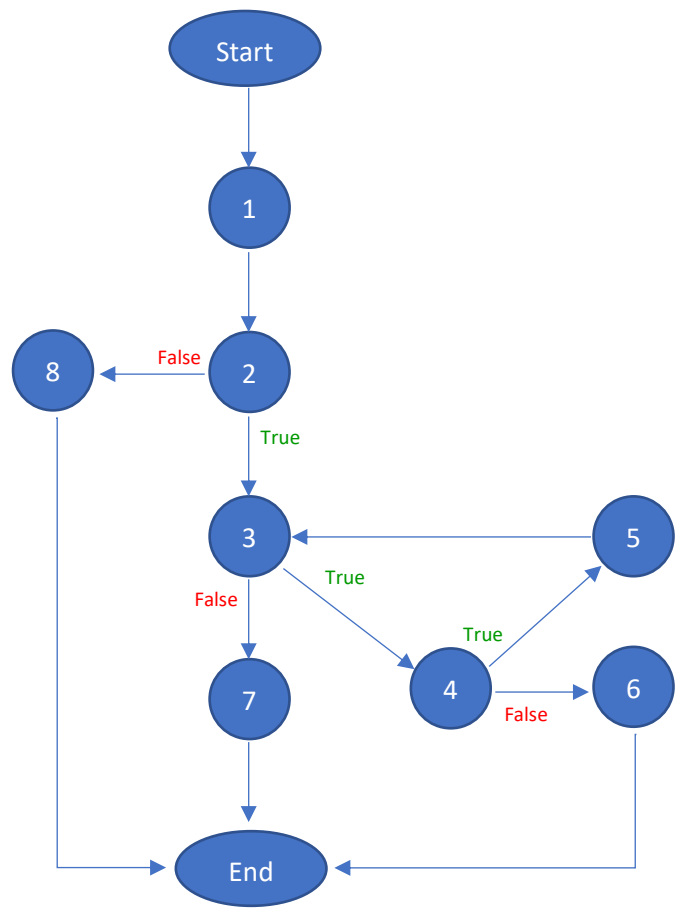
<pre>1 static boolean is_char_constant(String str) { 2 if (str.length() > 2 && str.charAt(0)=='#' && Character.isLetter(str.charAt(1))) 3 return true; 4 else 5 return false; }</pre>			
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	5	5	5



is_num_constant()

```
1 static boolean is_num_constant(String str)
2 {
3     if ( Character.isDigit(str.charAt(0)))
4     {
5         while ( i <= str.length() && str.charAt(i) != '\0' )    /* until meet
6             token end sign */
7             {
8                 if(Character.isDigit(str.charAt(i+1)))
9                 {
10                     i++;
11                 }
12                 else
13                 {
14                     return false;
15                 }
16             }
17             return true;
18         }
19     }
20     else
21     {
22         return false;
23     }
24 }
```

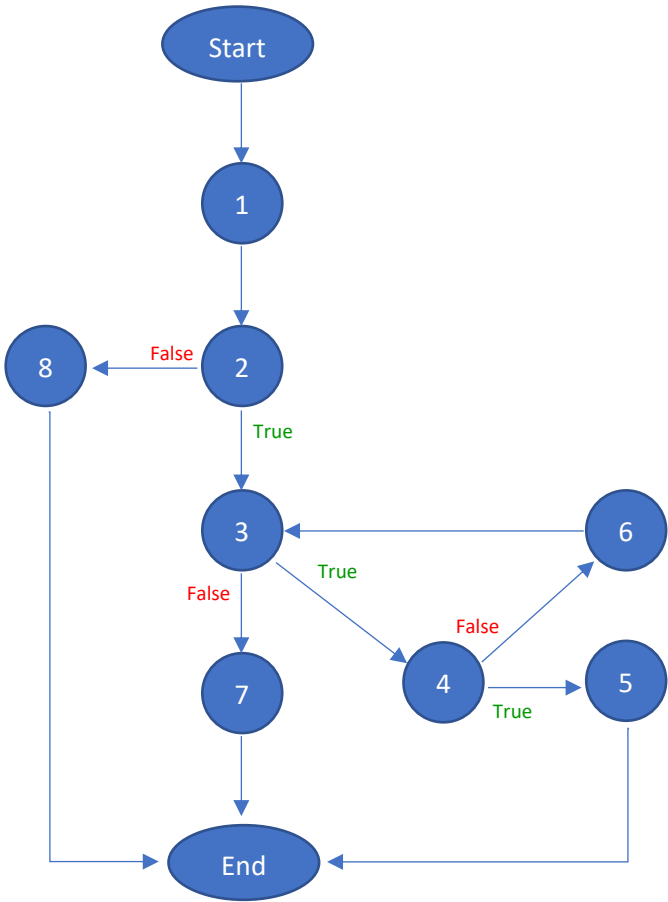
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



is_str_constant()

```
1 static boolean is_str_constant(String str)
2 {
3     int i=1;
4     if ( str.charAt(0) ==''')
5     { while (i < str.length() && str.charAt(0)!='\0') /* until meet the
6       token end sign */
7       { if(str.charAt(i)=='')
8         return true; /* meet the second ''' */
9       }
10      else
11      { i++; /* end WHILE */
12      }
13      return true;
14    }
15    else
16    { return false; /* other return FALSE */
17    }
18 }
```

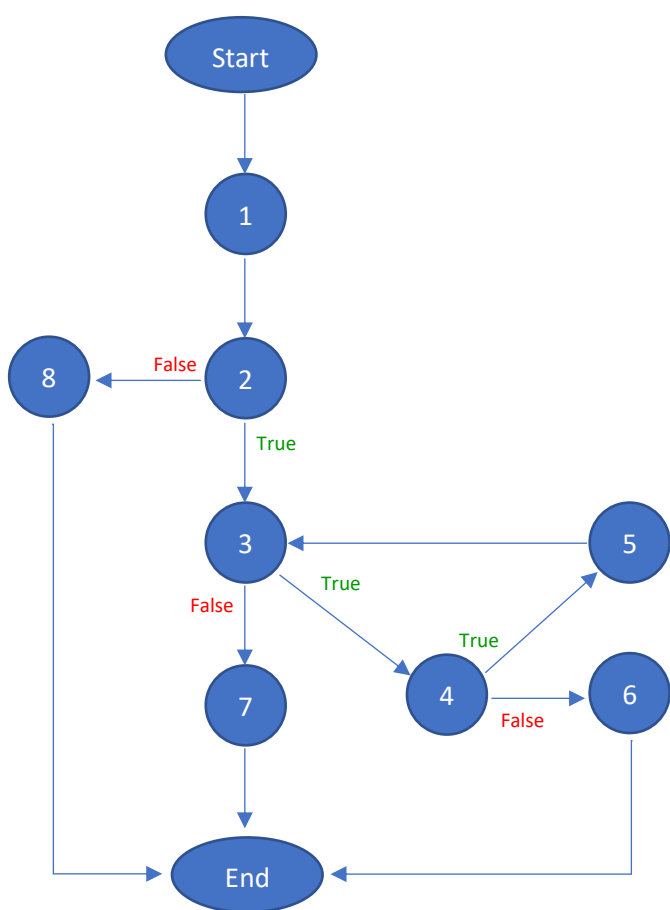
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



is_identifier()

```
1 static boolean is_identifier(String str)
2 {
3     int i=0;
4     if ( Character.isLetter(str.charAt(0)) )
5     {
6         while(i < str.length() && str.charAt(i) !='\0' ) /* until meet the
7         end token sign */
8         {
9             if(Character.isLetter(str.charAt(i)) ||
10             Character.isDigit(str.charAt(i)))
11             i++;
12         }
13         return false;
14     } /* end WHILE */
15     return false;
16 }
17 else
18 return true;
19 }
```

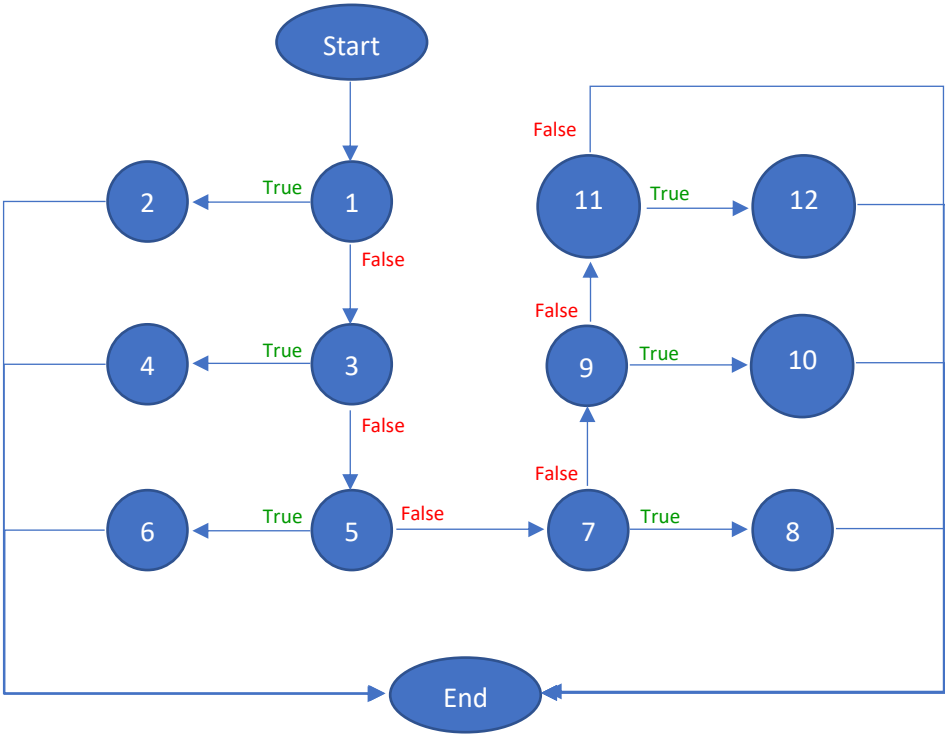
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



print_spec_symbol()

```
1 static void print_spec_symbol(String str) {
2     if (str.equals("{")) {
3         System.out.print("lparen.\n");
4         return;
5     }
6     if (str.equals("}")) {
7         System.out.print("rparen.\n");
8         return;
9     }
10    if (str.equals("[")) {
11        System.out.print("lsquare.\n");
12        return;
13    }
14    if (str.equals("]")) {
15        System.out.print("rsquare.\n");
16        return;
17    }
18    if (str.equals("'")) {
19        System.out.print("quote.\n");
20        return;
21    }
22    if (str.equals("`")) {
23        System.out.print("bquote.\n");
24        return;
25    }
26 }
```

Block	Lines	Entry	Exit
1	2	2	2
2	3, 4	3	4
3	5	5	5
4	6, 7	6	7
5	8	8	8
6	9, 10	9	10
7	11	11	11
8	12, 13	12	13
9	14	14	14
10	15, 16	15	16
11	17	17	17
12	18, 19	18	19



is_spec_symbol()

```
1 static boolean is_spec_symbol(char c)
2 {
3     if (c == '(')
4     {
5         return true;
6     }
7     if (c == ')')
8     {
9         return true;
10    }
11    if (c == '[')
12    {
13        return true;
14    }
15    if (c == ']')
16    {
17        return true;
18    }
19    if (c == '/')
20    {
21        return true;
22    }
23    if (c == '`')
24    {
25        return true;
26    }
27    if (c == ',')
28    {
29        return true;
30    }
31    return false; /* others return FALSE */
32 }
```

Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	7	7	7
7	8	8	8
8	9	9	9
9	10	10	10
10	11	11	11
11	12	12	12
12	13	13	13
13	14	14	14
14	15	15	15
15	16	16	16

