CSE 4321 Project
Control Flow Graph

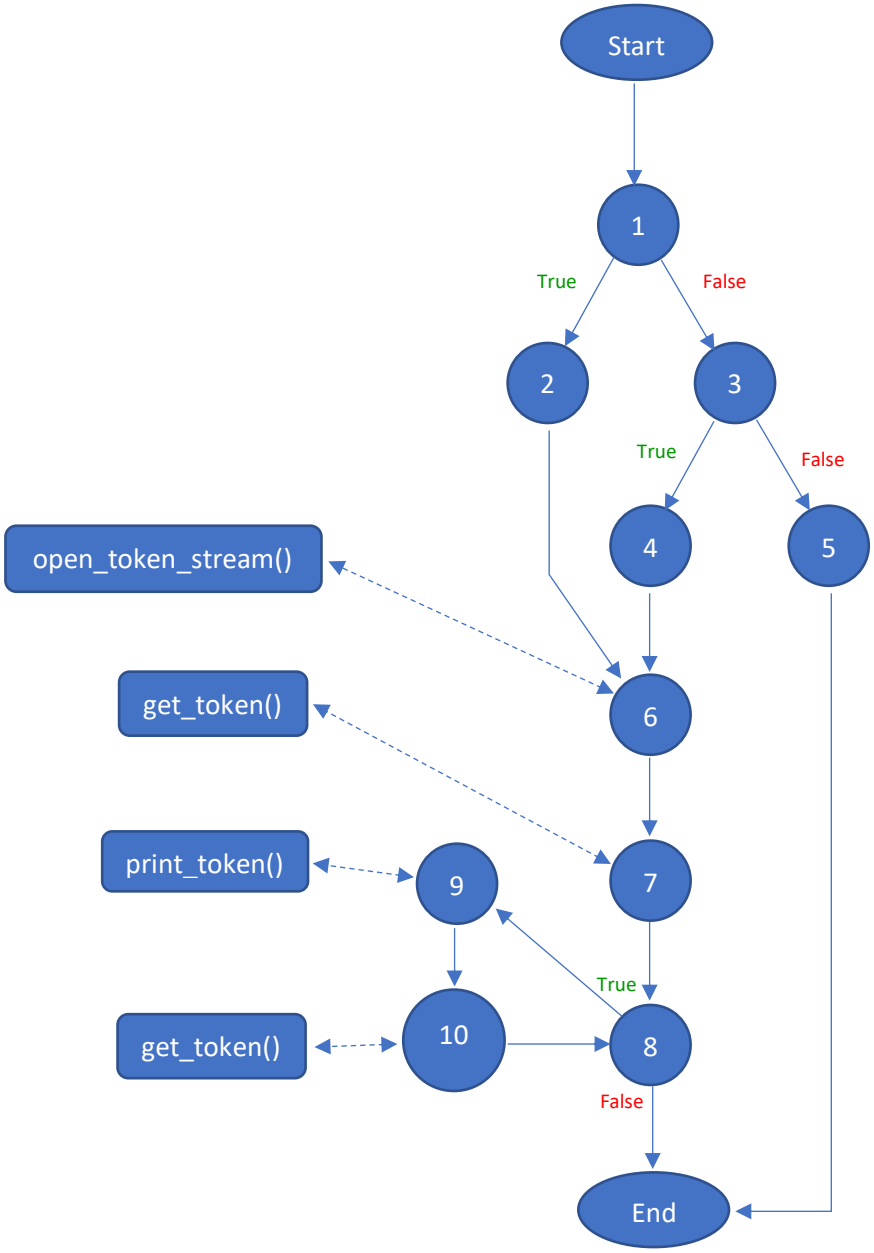
1000867697

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  Printtokens t = new Printtokens();
11  BufferedReader br = t.open_token_stream(fname);  /* open token stream
*/
12  String tok = t.get_token(br);
13  while (tok != null) {  /* take one token each time until eof */
14    t.print_token(tok);
15    tok = t.get_token(br);
    }

16     System.exit(0);
}
```

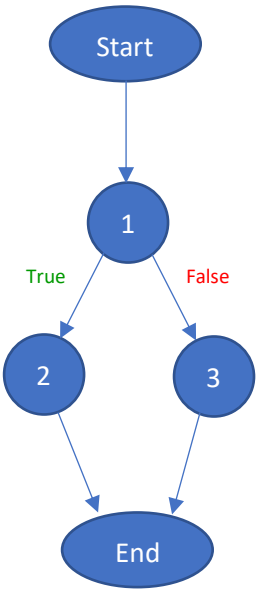
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	5	5	5
4	6	6	6
5	8, 9	8	9
6	10, 11	10	11
7	12	12	12
8	13	13	13
9	14	14	14
10	15	15	15



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  return null;
  }
```

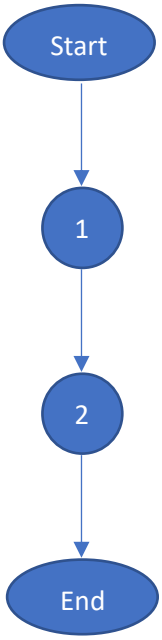
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	6, 7, 8, 9, 10, 11	6	11



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   return ch;
  }
```

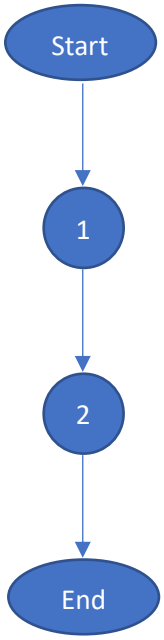
Block	Lines	Entry	Exit
1	2, 3, 4, 5, 6, 7	2	7
2	8	8	8



unget_char

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

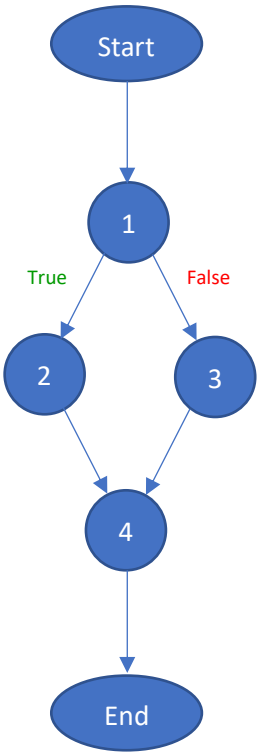
Block	Lines	Entry	Exit
1	2, 3, 4, 5	2	5
2	6	6	6



open_token_stream

```
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;
  }
```

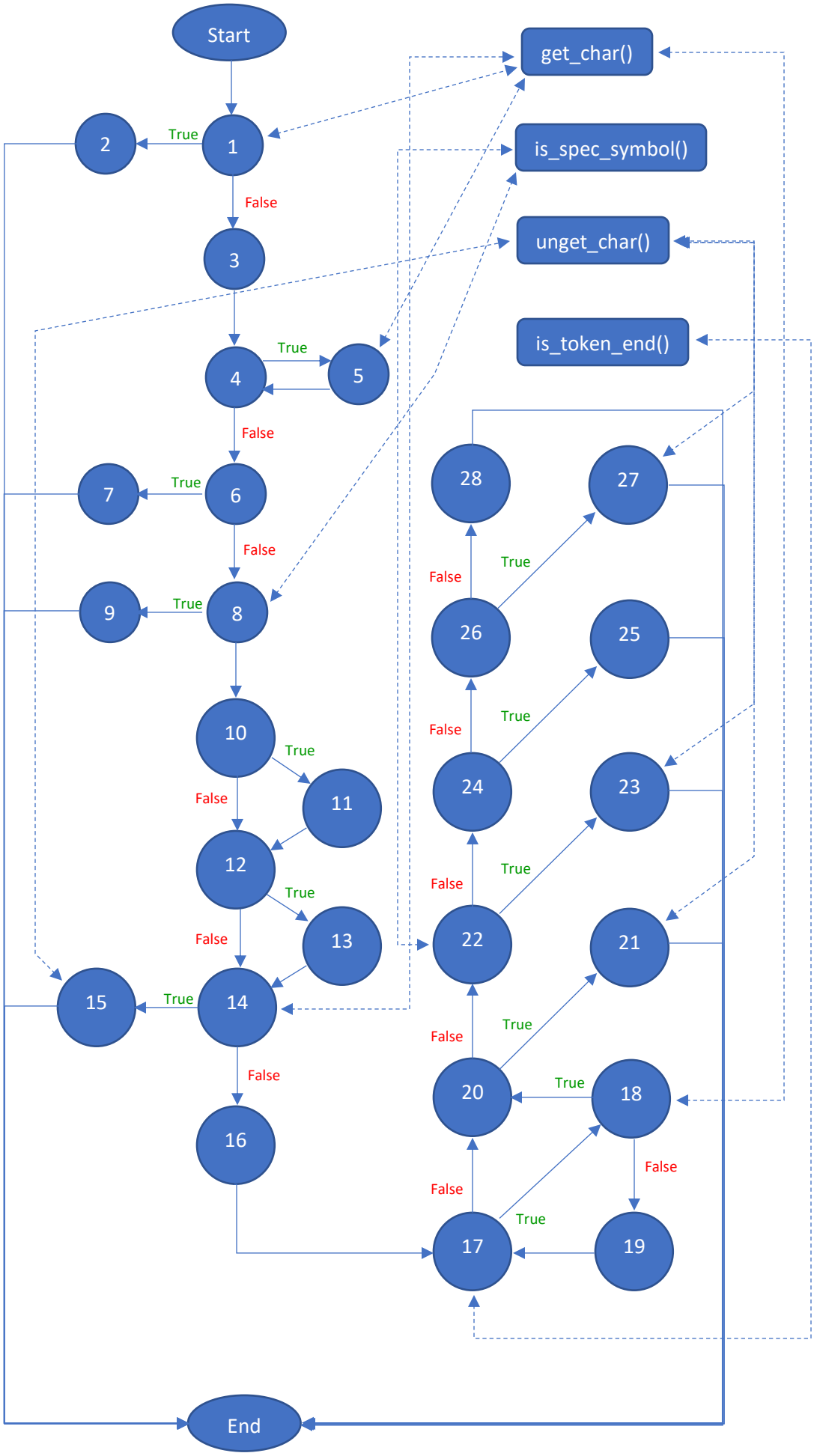
Block	Lines	Entry	Exit
1	2, 3	2	3
2	4	4	4
3	6	6	6
4	7	7	7



get_token

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

6    StringBuilder sb = new StringBuilder();

7    try {
8      res = get_char(br);
9      if (res == -1) {
10       return null;
       }
11     ch = (char)res;
12     while(ch=='\t'||ch=='\n' || ch == '\r')      /* strip all blanks
until meet characters */
       {
13       res = get_char(br);
14       ch = (char)res;
       }

15     if(res == -1) 16 return null;
17     sb.append(ch);
18     if(is_spec_symbol(ch)==true) 19 return sb.toString();
20     if(ch =='"') 21 id=2;     /* prepare for string */
22     if(ch ==59) 23 id=1;     /* prepare for comment */

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

29     while (is_token_end(id,res) == false)/* until meet the end
character */
       {
30       sb.append(ch);
31       br.mark(4);
32       res = get_char(br);
33       if (res == -1) {
34         break;
         }
35       ch = (char)res;
       }

36     if(res == -1)       /* if end character is eof token    */
37     { unget_char(ch,br);        /* then put back eof on token_stream */
38       return sb.toString();
       }

39     if(is_spec_symbol(ch)==true)      /* if end character is
special_symbol */
40     { unget_char(ch,br);        /* then put back this character
*/
41       return sb.toString();
       }
42     if(id==1)                  /* if end character is " and is string
*/
       {
43       sb.append(ch);
44       return sb.toString();
       }
45     if(id==0 && ch==59)
                                   /* when not in string or
comment,meet ";" */
46     { unget_char(ch,br);       /* then put back this character
*/
47       return sb.toString();
       }
48     } catch (IOException e) {
49       e.printStackTrace();
     }

50   return sb.toString();                    /* return nomal case token
*/
   }
```

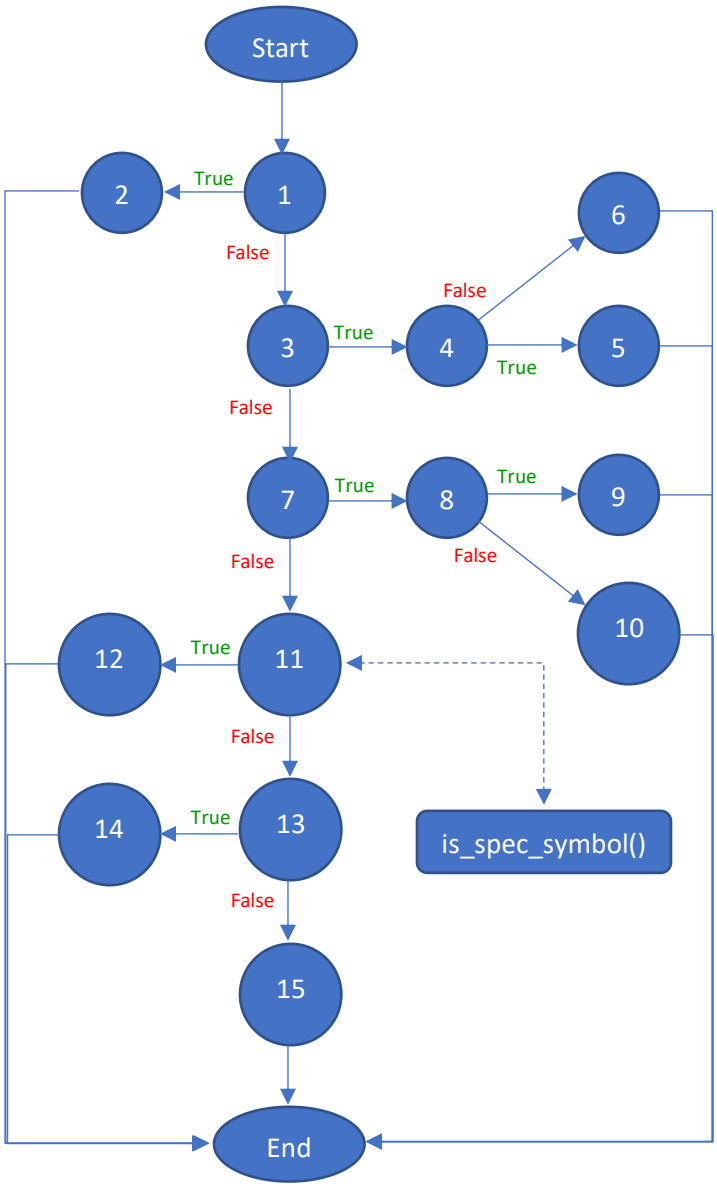
Block	Lines	Entry	Exit
1	2, 3, 4, 5, 6, 7, 8, 9	2	9
2	10	10	10
3	11	11	11
4	12	12	12
5	13, 14	13	14
6	15	15	15
7	16	16	16
8	17, 18	17	18
9	19	19	19
10	20	20	20
11	21	21	21
12	22	22	22
13	23	23	23
14	24, 25	24	25
15	26, 27	26	27
16	28	28	28
17	29	29	29
18	30, 31, 32, 33	30	33
19	35	35	35
20	36	36	36
21	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


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

is_token_end

```
1  static boolean is_token_end(int str_com_id, int res)
   {
2     if(res==-1) 3 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
another " */
7          return true;
8       else
9          return false;
      }

10    if(str_com_id==2)     /* is comment token */
11    { if(ch=='\n' || ch == '\r' || ch=='\t')      /* for comment until meet
end of line */
12         return true;
13      else
14         return false;
      }

15    if(is_spec_symbol(ch)==true) 16 return true; /* is special_symbol? */
17    if(ch ==' ' || ch=='\n'|| ch=='\r' || ch==59) 18 return true;

19    return false;                /* other case,return FALSE */
   }
```

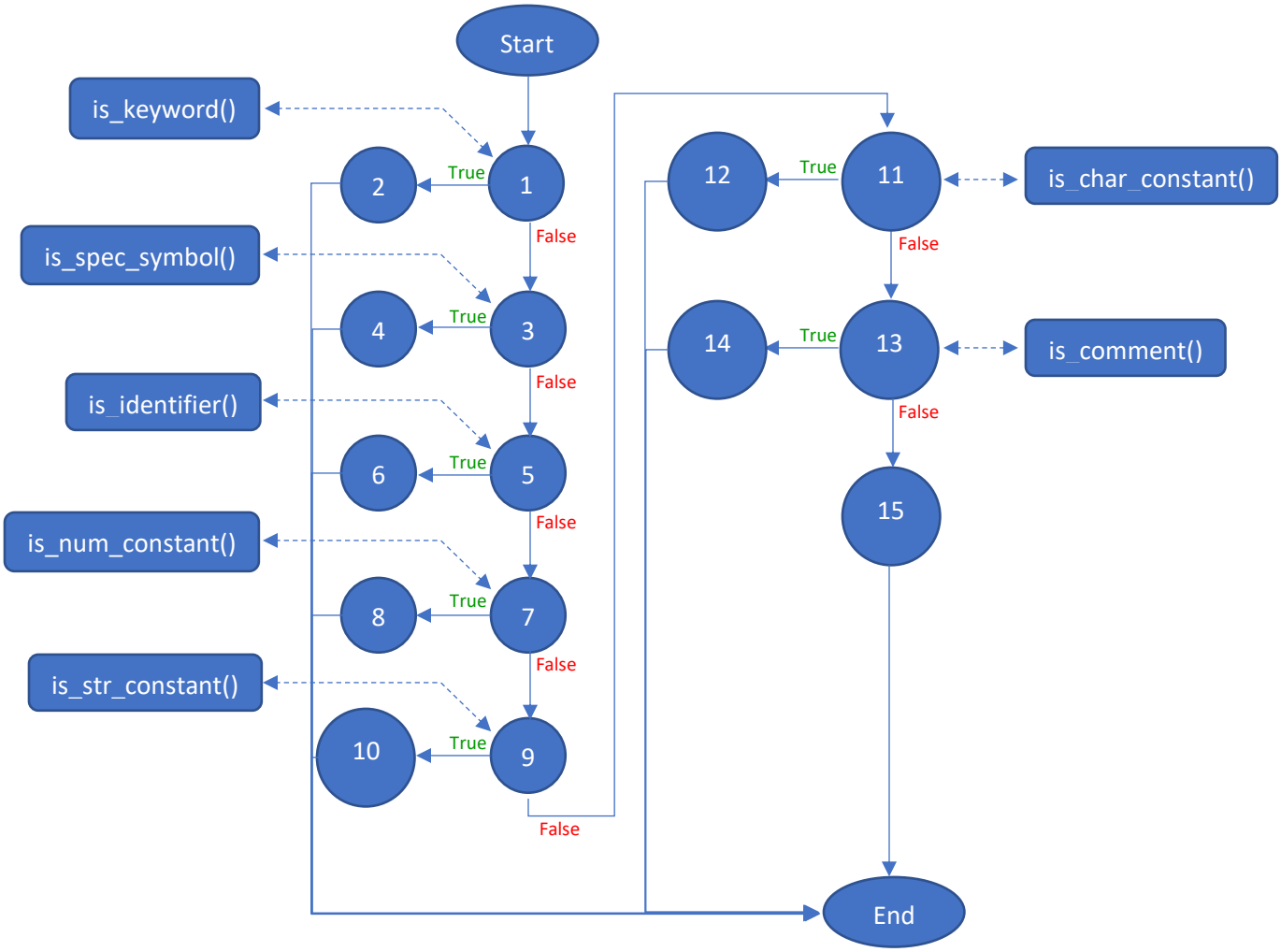
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4, 5	4	5
4	6	6	6
5	7	7	7
6	9	9	9
7	10	10	10
8	11	11	11
9	12	12	12
10	14	14	14
11	15	15	15
12	16	16	16
13	17	17	17
14	18	18	18
15	19	19	19



token_type

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

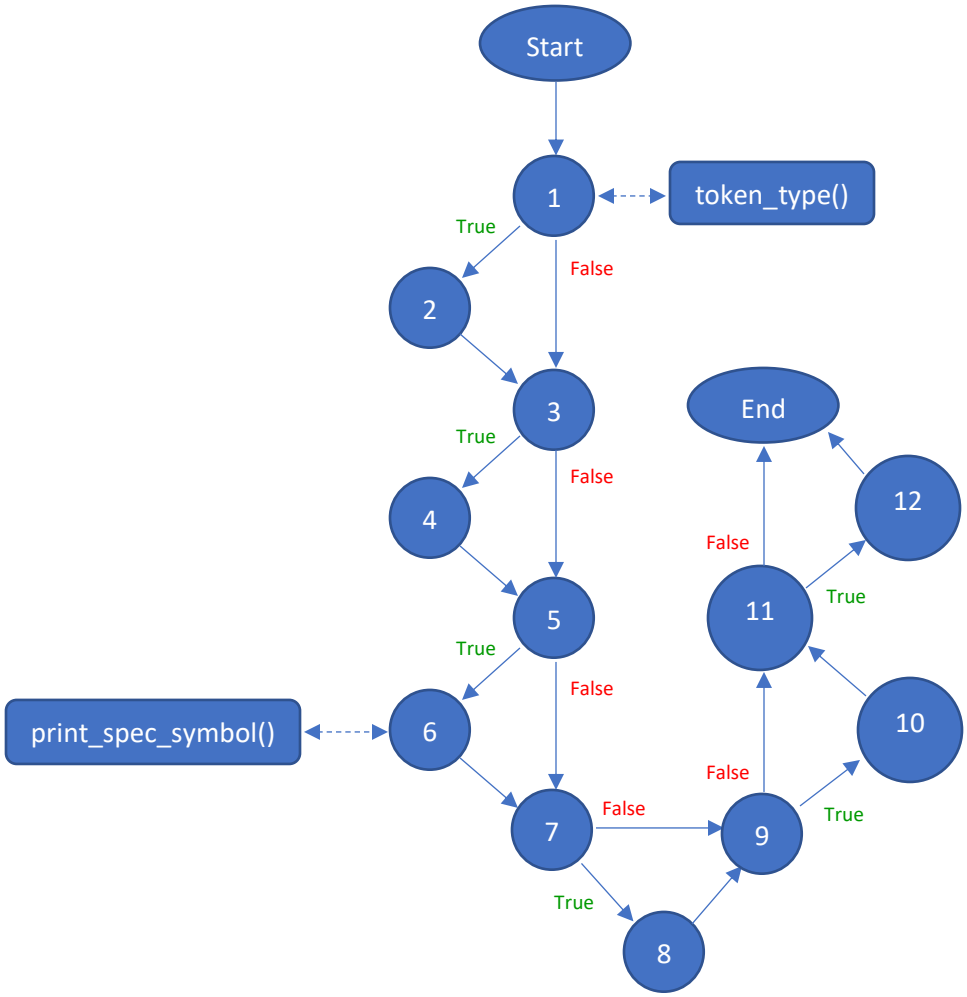
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	7	7	7
7	8	8	8
8	9	9	9
9	10	10	10
10	11	11	11
11	12	12	12
12	13	13	13
13	14	14	14
14	15	15	15
15	16	16	16



print_token

```
1  void print_token(String tok)
2  { int type;
3    type=token_type(tok);
4    if(type==error)
     {
6      System.out.print("error,\"" + tok + "\".\n");
     }

7    if(type==keyword)
     {
8      System.out.print("keyword,\"" + tok + "\".\n");
     }

9    if(type==spec_symbol) 10 print_spec_symbol(tok);
11   if(type==identifier)
     {
12     System.out.print("identifier,\"" + tok + "\".\n");
     }
13   if(type==num_constant)
     {
14     System.out.print("numeric," + tok + ".\n");
     }
15   if(type==char_constant)
     {
16     System.out.print("character,\"" + tok.charAt(1) + "\".\n");
     }

   }
```

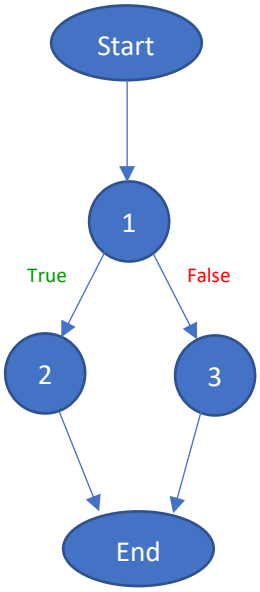
Block	Lines	Entry	Exit
1	2, 3, 4	2	4
2	6	6	6
3	7	7	7
4	8	8	8
5	9	9	9
6	10	10	10
7	11	11	11
8	12	12	12
9	13	13	13
10	14	14	14
11	15	15	15
12	16	16	16



is_comment

```
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;
  }
```

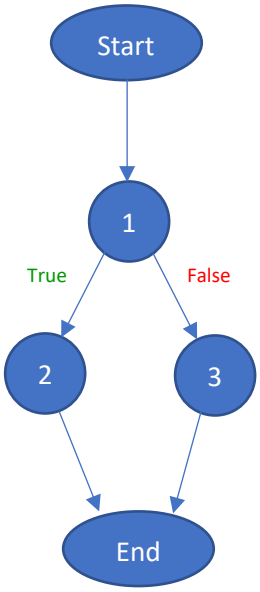
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	5	5	5



is_keyword

```
1 static boolean is_keyword(String str)
  {
2   if (str.equals("and") || str.equals("or") || str.equals("if") ||
        str.equals("xor")||str.equals("lambda")||str.equals("=>"))
3     return true;
4   else
5     return false;
  }
```

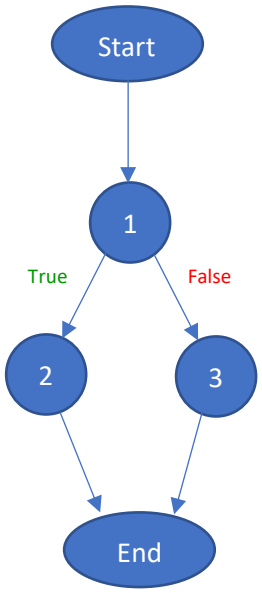
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	5	5	5



is_char_constant

```
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;
  }
```

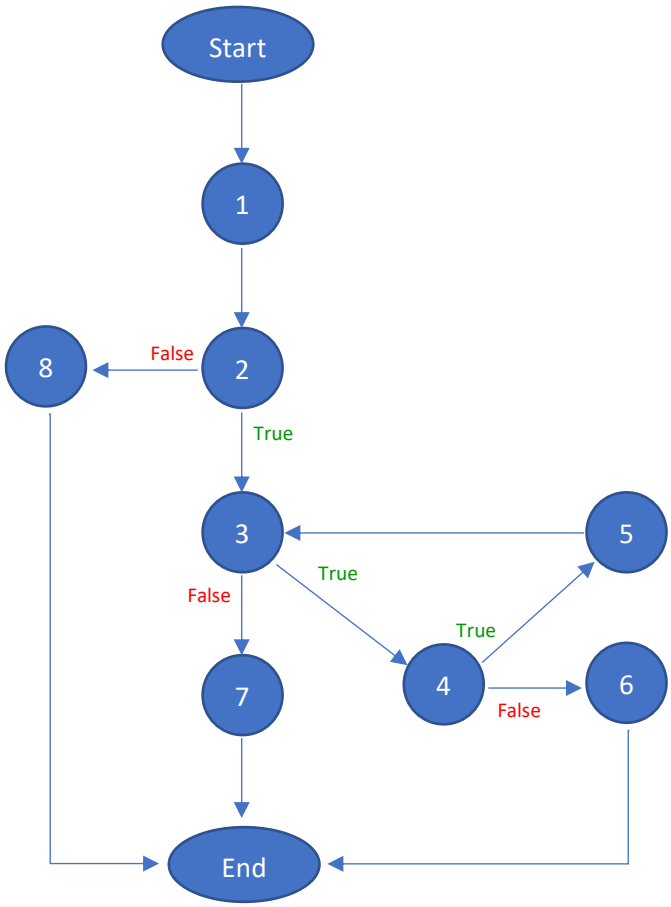
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	5	5	5



is_num_constant

```
1  static boolean is_num_constant(String str)
   {
2    int i=1;

3    if ( Character.isDigit(str.charAt(0)))
     {
4      while ( i <= str.length() && str.charAt(i) != '\0' )    /* until meet
token end sign */
       {
5        if(Character.isDigit(str.charAt(i+1)))
6          i++;
7        else
8          return false;
       }                              /* end WHILE */
9      return true;
     }
10   else
11     return false;                  /* other return FALSE */
   }
```

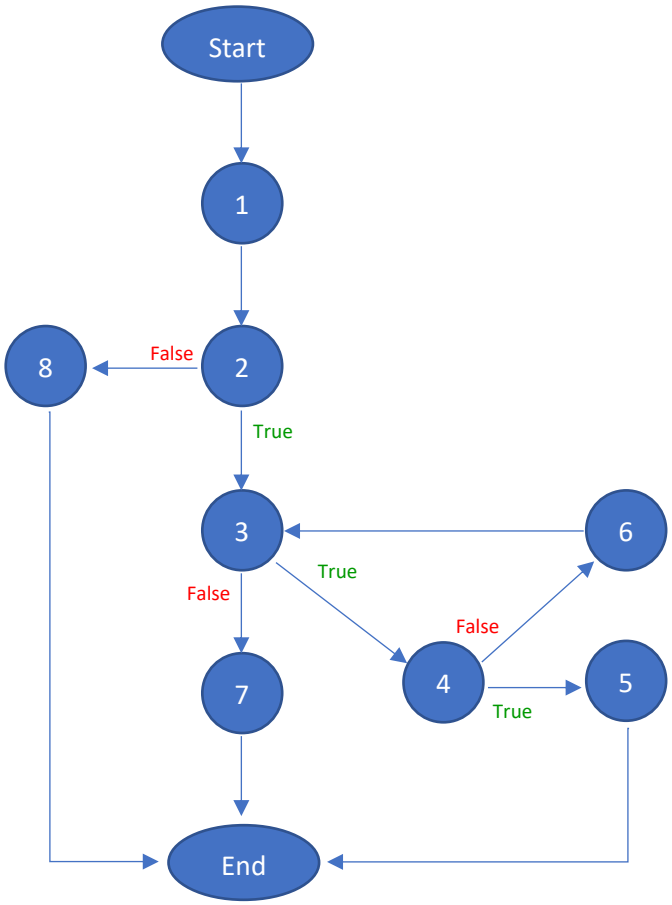
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



is_str_constant

```
1  static boolean is_str_constant(String str)
   {
2    int i=1;

3    if ( str.charAt(0) =='"')
4    { while (i < str.length() && str.charAt(0)!='\0')  /* until meet the
token end sign */
5      { if(str.charAt(i)=='"')
6          return true;         /* meet the second '"'           */
7        else
8          i++;
       }                  /* end WHILE */
9      return true;
     }
10   else
11     return false;        /* other return FALSE */
   }
```

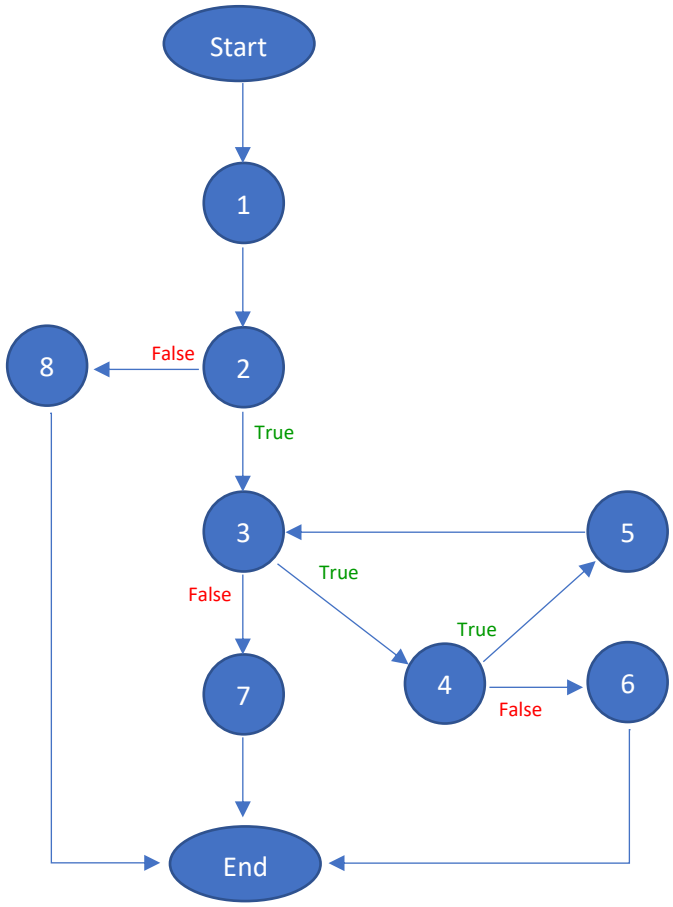
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



is_identifier

```
1  static boolean is_identifier(String str)
   {
2     int i=0;

3     if ( Character.isLetter(str.charAt(0)) )
      {
4        while(i < str.length() && str.charAt(i) !='\0' )    /* unti meet the
end token sign */
         {
5           if(Character.isLetter(str.charAt(i)) ||
Character.isDigit(str.charAt(i)))
6              i++;
7           else
8              return false;
         }      /* end WHILE */
9        return false;
      }
10    else
11       return true;
}
```

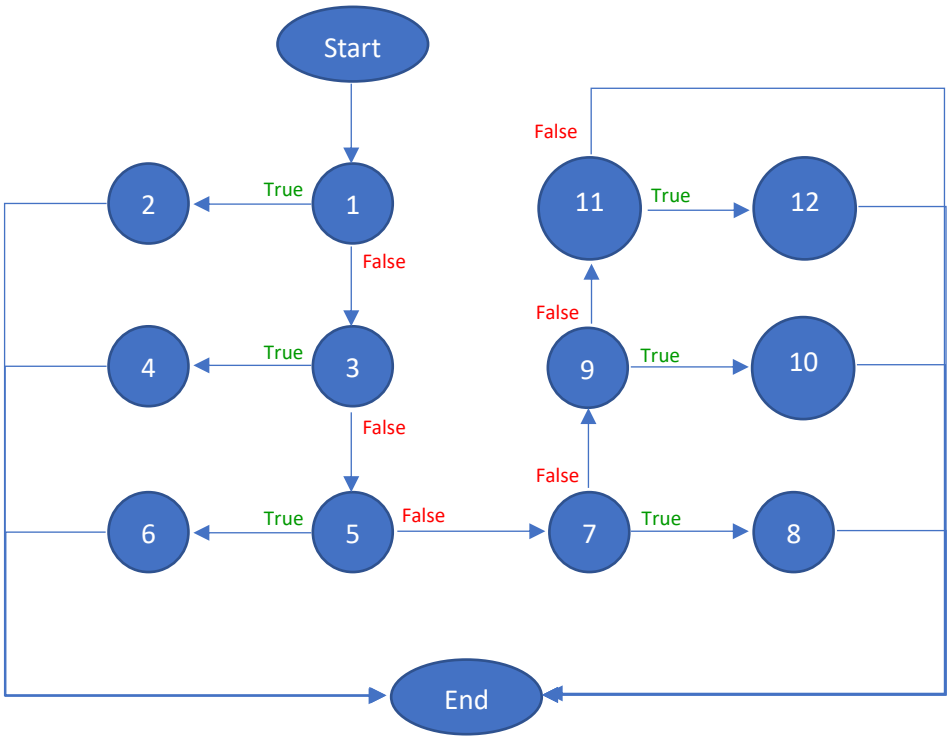
Block	Lines	Entry	Exit
1	2	2	2
2	3	3	3
3	4	4	4
4	5	5	5
5	6	6	6
6	8	8	8
7	9	9	9
8	11	11	11



print_spec_symbol

```
1  static void print_spec_symbol(String str) {
2      if (str.equals("{")) {

3          System.out.print("lparen.\n");
4          return;
       }
5      if (str.equals(")")) {

6          System.out.print("rparen.\n");
7          return;
       }
8      if (str.equals("[")) {
9          System.out.print("lsquare.\n");
10         return;
       }
11     if (str.equals("]")) {

12         System.out.print("rsquare.\n");
13         return;
       }
14     if (str.equals("'")) {
15         System.out.print("quote.\n");
16         return;
       }
17     if (str.equals("`")) {

18         System.out.print("bquote.\n");
19         return;
   }


}
```

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     if (c == '(')
      {
3        return true;
      }
4     if (c == ')')
      {
5        return true;
      }
6     if (c == '[')
      {
7        return true;
      }
8     if (c == ']')
      {
9        return true;
      }
10    if (c == '/')
      {
11       return true;
      }
12    if (c == '`')
      {
13       return true;
      }
14    if (c == ',')
      {
15       return true;
      }
16    return false;      /* others return FALSE */
   }
```

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