SYSC 4101

Laboratory 6

Exercise 1 (10 marks)

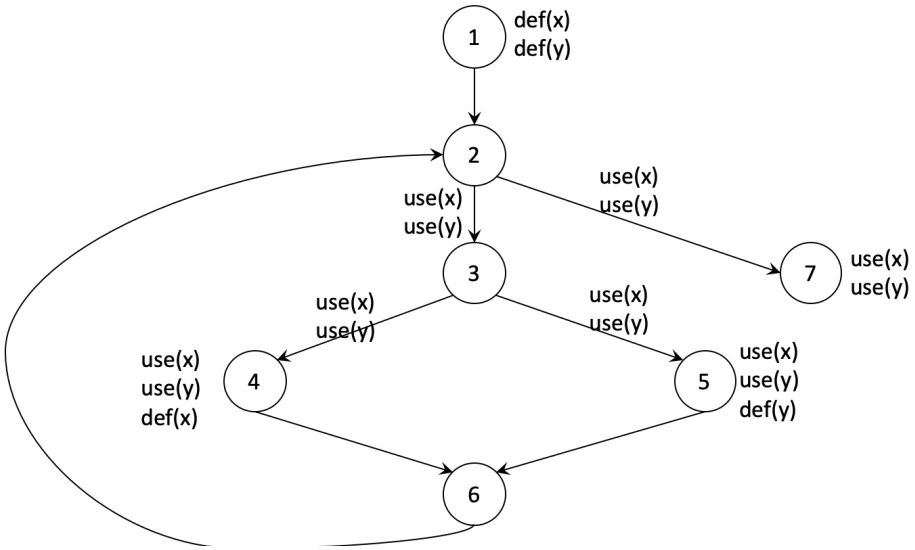
Create the control flow graph for the following two Java methods: use the condensed form; label control flow nodes after line number(s); label branches with Y or N depending on conditions.

```
1   protected static final String addEscapes(String str) {
2      StringBuilder retval = new StringBuilder();
3      char ch;
4      for (int i = 0; i < str.length(); i++) {
5         switch (str.charAt(i)) {
6            case '\b':
7               retval.append("\\b");
8               continue;
9            case '\t':
10              retval.append("\\t");
11              continue;
12           case '\n':
13              retval.append("\\n");
14              continue;
15           case '\f':
16              retval.append("\\f");
17              continue;
18           case '\r':
19              retval.append("\\r");
20              continue;
21           case '\"':
22              retval.append("\\\"");
23              continue;
24           case '\'':
25              retval.append("\\\'");
26              continue;
27           case '\\':
28              retval.append("\\\\");
29              continue;
30           default:
31              if ((ch = str.charAt(i)) < 0x20 || ch > 0x7e) {
32                 String s = "0000" + Integer.toString(ch, 16);
33                 retval.append("\\u" + s.substring(s.length() - 4, s.length()));
34              } else {
35                 retval.append(ch);
36              }
37              continue;
38        }
39     }
40     return retval.toString();
41  }
```

```
1   void foo() {
2       Random r = new Random();
3       Set<Integer> aSet= new HashSet<Integer>();
4       int anInt;
5       do {
6           anInt = r.nextInt(10);
7           if (anInt % 2 == 0)
8               continue;
9           System.out.println(anInt);
10      } while (aSet.add(anInt));
11      System.out.println(aSet);
12  }
```

Exercise 2 (50 marks)

Consider the graph below: it has 7 nodes and 8 edges; it has one initial node, namely node 1, and one final node, namely node 7. Nodes and edges are labeled with data flow information, specifically uses and definitions of variables x and y. The order of appearance (top to bottom) of uses and definitions in labels for nodes 4 and 5 indicates the actual order of executions of the uses and definitions at those nodes.



Question 1: [5 marks]
List the round trip paths for this graph.

Question 2: [5 marks]
List the test requirements (a.k.a. test objectives) for the Simple-Round-Trip criterion.

Question 3: [5 marks]
Create a set of test paths, i.e., paths from the initial node to the final node, that is adequate for the Simple-Round-Trip criterion.

Question 4: [5 marks]
Is the Simple-Round-Trip adequate test suite you created in the previous question adequate for the Complete-Round-Trip criterion? Justify.

Question 5: [5 marks]
List the test requirements (a.k.a. test objectives) for the All-Defs criterion. (Remember to account for both variables x and y.)

Question 6: [5 marks]
Create a set of test paths, i.e., paths from the initial node to the final node, that is adequate for the All-Defs criterion.

Question 7: [5 marks]
List the test requirements (a.k.a. test objectives) for the All-Uses criterion. (Remember to account for both variables x and y.)

Question 8: [5 marks]
Create a set of test paths, i.e., paths from the initial node to the final node, that is adequate for the All-Uses criterion.

Question 9: [5 marks]
List the test requirements (a.k.a. test objectives) for the All-DU-paths criterion. (Remember to account for both variables x and y.)

Question 10: [5 marks]
Create a set of test paths, i.e., paths from the initial node to the final node, that is adequate for the All-DU-paths criterion.